Balancing Learning Regimes: The Impact of Prior Knowledge on the Dynamics of Neural Representations

Clémentine Carla Juliette Dominé

A dissertation submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

of

University College London.

Gatsby Computational Neuroscience Unit

University College London

July 17, 2025

I, Clémentine Carla Juliette Dominé, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Abstract

Biological and artificial neural networks create internal representations that support complex tasks like reasoning and decision-making. While characterising these representations has been a key focus in machine learning and neuroscience, the mechanisms for extracting task-specific features and the influence of prior knowledge remain unclear. To gain insight, it's essential to analyse the interactions between datasets, network architectures and initialisation. Previous work has shown that specific initialisations place networks in a 'lazy' regime, where internal representations do not change through learning, while other initialisations place networks in a 'rich/feature learning' regime, where internal representations evolve to fit a task. Here, we study how initialisation and architecture learning structured data influence learning dynamics in deep neural networks. First, we derive novel exact solutions for deep linear networks with 'lambda-balanced' initialisations that differ in the norms of the weights across layers, which approximate common initialisations used in practice. Our results show that imbalanced initialisations lead to a lazy learning regime, while balanced ones promote a rich regime. These findings enhance the understanding of how weight initialisation and network structure influence learning, with implications for continual, reversal, and transfer learning in neuroscience and practical applications. Next, we demonstrate that our theoretical findings, derived from deep linear networks, have significant implications for non-linear networks. Utilising the non-linear teacher-student theoretical framework for neural network analysis, we reveal a strong dependence of specialisation—characterised by rich, task-specific representations—on initial weight imbalance. We discuss the implications of this understanding in the context of continual learning and showcase its application in practical machine learning scenarios, such as grokking, developing edge detectors in convolutional neural networks, and neuroscience. Overall, our results highlight the critical role of initialisation imbalance in the learning dynamics of both artificial and biological neural networks.

Impact Statement

The expertise, knowledge, and analytical frameworks developed in this thesis have led to significant contributions across deep learning theory, machine learning, cognitive science, and neuroscience. This research not only advances fundamental understanding in these fields but also holds substantial potential to enhance interoperability within the machine learning community and promote the safe deployment of AI systems. Furthermore, this research has the potential to inform strategies for optimizing efficient feature learning.

Concretely, this work has resulted in six conference papers published in top-tier machine learning venues [39, 69, 67, 219, 158, 203, 134]. Other works were conducted as part of this doctoral research but are not directly included in this thesis due to considerations of relevance and space [275, 68, 95, 214]. These papers have been widely recognized by the international research community, cited by peers, and presented at prestigious conferences, including NeurIPS, ICML and ICLR, as well as in invited talks at international, national, and regional levels. Notably, citations from both the neuroscience and machine learning communities underscore the interdisciplinary impact of this research.

Beyond my individual contributions, I am actively engaged in fostering collaboration within the academic community. I co-organize the UniReps: Unifying Representations in Neural Models workshop at NeurIPS, an initiative dedicated to bridging insights between biological and artificial neural networks—precisely the domain in which this thesis is situated.

This work has already spurred further advancements, with several projects emerging as direct follow-ups. Moreover, the findings presented here establish a robust foundation for future research, supported by key sponsors such as the Bilal Cluster of Excellence and the Harvard NTT Scholarship.

UCL Research Paper Declaration Form: Referencing the Doctoral Candidate's Own Published Work(s)

1. For a research manuscript that has already been published (if not yet published, please skip to section 2):

(a) **What is the title of the manuscript?**
Exact learning dynamics of deep linear networks with prior knowledge

(b) **Please include a link to or DOI for the work:**
NeurIPS 2022 Paper

(c) **Where was the work published?**
Advances in Neural Information Processing Systems 35 (NeurIPS 2022) Main Conference Track

(d) **Who published the work?**
Advances in Neural Information Processing Systems 35

(e) **When was the work published?**
Dec 6 2022

(f) **List the manuscript's authors in the order they appear on the publication:**
Braun, Lukas*, Clémentine Dominé*, James Fitzgerald, and Andrew Saxe.
*denotes the co-first authorship.

(g) **Was the work peer reviewed?**
Yes

(h) **Have you retained the copyright?**
Yes

(i) **Was an earlier form of the manuscript uploaded to a preprint server (e.g. medRxiv)? If 'Yes', please give a link or DOI**
No
If 'No', please seek permission from the relevant publisher and check the box

below:

✓ I acknowledge permission of the publisher named under 1d to include in this thesis portions of the publication named as included in 1c.

2. For a research manuscript prepared for publication but that has not yet been published (if already published, please skip to section 3):

(a) **What is the current title of the manuscript?**

(b) **Has the manuscript been uploaded to a preprint server (e.g. medRxiv)?** If 'Yes', please give a link or DOI:

(c) **Where is the work intended to be published?**

(d) **List the manuscript's authors in the intended authorship order:**

(e) **Stage of publication:**

3. For multi-authored work, please give a statement of contribution covering all authors (if single-author, please skip to section 4):

Lukas Braun and Clémentine C. J. Domine, contributed to the derivation of the solutions and to the application to the rich and lazy learning regime. Lukas focused on the dynamics of continual learning and on extending the results to handle multiple outputs. Additional findings on the dynamics of alignment, reversal learning, and knowledge revision -developed by Clémentine C. J. Domine - are detailed in the original publication. James Fitzgerald and Andrew Saxe were involved in a supervisory role throughout the project. All authors contributed to the writing of the appendix and the polishing of the manuscript.

4. In which chapter(s) of your thesis can this material be found?

Chapter 1, 2, 3

e-Signatures confirming that the information above is accurate

This form should be co-signed by the supervisor/senior author unless this is not appropriate (e.g. if the paper was a single-author work):

Candidate: Clementine Domine	**Date:** 14/04/2025
Supervisor/Senior Author: Andrew Saxe	**Date:** May 8 2025

UCL Research Paper Declaration Form: Referencing the Doctoral Candidate's Own Published Work(s)

1. For a research manuscript that has already been published (if not yet published, please skip to section 2):

(a) **What is the title of the manuscript?**
Exact learning dynamics of deep linear networks with prior knowledge

(b) **Please include a link to or DOI for the work:**
DOI 10.1088/1742-5468/ad01b8

(c) **Where was the work published?**
Journal of Statistical Mechanics: Theory and Experiment, Volume 2023, November 2023

(d) **Who published the work?**
Published on behalf of SISSA Medialab srl by IOP Publishing Ltd

(e) **When was the work published?**
15 November 2023

(f) **List the manuscript's authors in the order they appear on the publication:**
Braun, Lukas*, Clémentine Dominé*, James Fitzgerald, and Andrew Saxe. where * denotes the co-first authorship.

(g) **Was the work peer reviewed?**
Yes

(h) **Have you retained the copyright?**
Yes

(i) **Was an earlier form of the manuscript uploaded to a preprint server (e.g. medRxiv)? If 'Yes', please give a link or DOI**
No

If ‘No’, please seek permission from the relevant publisher and check the box below:

✓ I acknowledge permission of the publisher named under 1d to include in this thesis portions of the publication named as included in 1c.

2. For a research manuscript prepared for publication but that has not yet been published (if already published, please skip to section 3):

(a) **What is the current title of the manuscript?**

(b) **Has the manuscript been uploaded to a preprint server (e.g. medRxiv)?** If ‘Yes’, please give a link or DOI:

(c) **Where is the work intended to be published?**

(d) **List the manuscript’s authors in the intended authorship order:**

(e) **Stage of publication:**

3. For multi-authored work, please give a statement of contribution covering all authors (if single-author, please skip to section 4):

Lukas Braun and Clémentine C. J. Domine, contributed to the derivation of the solutions and the application to the rich and lazy learning regime. Lukas focused on the dynamics of continual learning and on extending the results to handle multiple outputs. Additional findings on the dynamics of alignment, reversal learning, and knowledge revision - developed by Clémentine C. J. Domine — are detailed in the original publication. James Fitzgerald and Andrew Saxe were involved in a supervisory role throughout the project. All authors contributed to the writing of the appendix and the polishing of the manuscript.

4. In which chapter(s) of your thesis can this material be found?

Chapter 1, 2, 3

e-Signatures confirming that the information above is accurate

This form should be co-signed by the supervisor/senior author unless this is not appropriate (e.g. if the paper was a single-author work):

Candidate: Clementine Domine	**Date:** 14/ 04/2025
Supervisor/Senior Author: Andrew Saxe	**Date:** May 8 2025

UCL Research Paper Declaration Form: Referencing the Doctoral Candidate's Own Published Work(s)

1. For a research manuscript that has already been published (if not yet published, please skip to section 2):

(a) **What is the title of the manuscript?**
Get rich quick: exact solutions reveal how unbalanced initializations promote rapid feature learning

(b) **Please include a link to or DOI for the work:**
NeurIPS 2024 Paper

(c) **Where was the work published?**
Advances in Neural Information Processing Systems 37 (NeurIPS 2024) Main Conference Track

(d) **Who published the work?**
Advances in Neural Information Processing Systems 37

(e) **When was the work published?**
December 16 2024

(f) **List the manuscript's authors in the order they appear on the publication:**

Daniel Kunin*, Allan Raventós*, Clémentine Dominé, Feng Chen, David Klindt, Andrew Saxe, Surya Ganguli
$*$ denotes the co-first authorship.

(g) **Was the work peer reviewed?**
Yes

(h) **Have you retained the copyright?**
Yes

(i) **Was an earlier form of the manuscript uploaded to a preprint server (e.g. medRxiv)? If 'Yes', please give a link or DOI**
Yes DOI

✓ I acknowledge permission of the publisher named under 1d to include in this thesis portions of the publication named as included in 1c.

2. For a research manuscript prepared for publication but that has not yet been published (if already published, please skip to section 3):

(a) **What is the current title of the manuscript?**

(b) **Has the manuscript been uploaded to a preprint server (e.g. medRxiv)?**
If 'Yes', please give a link or DOI:

(c) **Where is the work intended to be published?**

(d) **List the manuscript's authors in the intended authorship order:**

(e) **Stage of publication:**

3. For multi-authored work, please give a statement of contribution covering all authors (if single-author, please skip to section 4):

This project originated from conversations between Daniel and Allan at the Kavli Institute for Theoretical Physics. Daniel, Allan, and Feng are primarily responsible for the single neuron analysis in Section 3. Daniel, Clem, Allan, and Feng are primarily responsible for the wide and deep linear analysis in Section 4. Daniel is primarily responsible for the nonlinear analysis in Section 5. Allan, Feng, and David are primarily responsible for the empirics in Fig. 1 and Fig. 6. Daniel is primarily responsible for writing the main sections. All authors contributed to the writing of the appendix and the polishing of the manuscript.

4. In which chapter(s) of your thesis can this material be found?
Chapter 2, 3 and 4

e-Signatures confirming that the information above is accurate

This form should be co-signed by the supervisor/senior author unless this is not appropriate (e.g. if the paper was a single-author work):

Candidate: Clementine Domine	**Date:** 14/ 04/2025
Supervisor/Senior Author: Andrew Saxe	**Date:** May 8 2025

UCL Research Paper Declaration Form: Referencing the Doctoral Candidate's Own Published Work(s)

1. For a research manuscript that has already been published (if not yet published, please skip to section 2):

(a) **What is the title of the manuscript?**

From lazy to rich: Exact learning dynamics in deep linear networks

(b) **Please include a link to or DOI for the work:**

DOI

(c) **Where was the work published?**

International Conference on Learning Representations (ICLR 2025) Main Comference Track

(d) **Who published the work?**

International Conference on Learning Representations (ICLR 2025)

(e) **When was the work published?**

9 Apr 2025

(f) **List the manuscript's authors in the order they appear on the publication:**

Clémentine Carla Juliette Dominé*, Nicolas Anguita*, Alexandra Maria Proca, Lukas Braun, Daniel Kunin, Pedro A. M. Mediano, Andrew M Saxe $*$ denotes the co-first authorship.

(g) **Was the work peer reviewed?**

Yes

(h) **Have you retained the copyright?**

Yes

(i) **Was an earlier form of the manuscript uploaded to a preprint server (e.g. medRxiv)? If 'Yes', please give a link or DOI**
DOI

✓ I acknowledge permission of the publisher named under 1d to include in this thesis portions of the publication named as included in 1c.

2. For a research manuscript prepared for publication but that has not yet been published (if already published, please skip to section 3):

(a) **What is the current title of the manuscript?**

(b) **Has the manuscript been uploaded to a preprint server (e.g. medRxiv)?**
If 'Yes', please give a link or DOI:

(c) **Where is the work intended to be published?**

(d) **List the manuscript's authors in the intended authorship order:**

(e) **Stage of publication:**

3. For multi-authored work, please give a statement of contribution covering all authors (if single-author, please skip to section 4):

Clementine and Nicolas led the work on the equal input-output solution to the exact dynamics, with support from Daniel. Clementine lead the extention to this work to the unequal input-output case from priminilary work from Nicolas. Then she examining the impact of the architecture. Clementine and Nicolas also spearheaded the development of the minimal model for *rich* and *lazy* learning, emphasizing exact solutions and representation convergence. Daniel, Nicolas, and Clementine collaborated on analyzing the dynamics of singular values, while Lukas led the investigation into representation robustness and sensitivity to noise. Clementine and Nicolas primarily conducted the analysis of the continual learning framework, while Alexandra and Clementine focused on transfer learning in the *rich* and *lazy* settings. Clementine took the lead on the reversal learning analysis. She was primarily responsible for leading the manuscript writing, with assistance from Daniel. All authors contributed to writing the appendix and refining the final manuscript.

4. In which chapter(s) of your thesis can this material be found?

Chapter 2 and 3

e-Signatures confirming that the information above is accurate

This form should be co-signed by the supervisor/senior author unless this is not appropriate (e.g. if the paper was a single-author work):

Candidate: Clementine Domine	**Date:** 14/ 04/2025
Supervisor/Senior Author: Andrew Saxe	**Date:** May 8 2025

UCL Research Paper Declaration Form: Referencing the Doctoral Candidate's Own Published Work(s)

1. For a research manuscript that has already been published (if not yet published, please skip to section 2):

(a) **What is the title of the manuscript?**
A Theory of Initialisation's Impact on Specialisation

(b) **Please include a link to or DOI for the work:**
https://arxiv.org/abs/2503.02526

(c) **Where was the work published?**
International Conference on Learning Representations (ICLR 2025) Main Comference Track

(d) **Who published the work?**
International Conference on Learning Representations (ICLR 2025)

(e) **When was the work published?**
9 Apr 2025

(f) **List the manuscript's authors in the order they appear on the publication:**
Devon Jarvis*, Sebastian Lee*, Clémentine Carla Juliette Dominé*, Andrew M Saxe, Stefano Sarao Mannelli *denotes the co-first authorship.

(g) **Was the work peer reviewed?**
Yes

(h) **Have you retained the copyright?**
Yes

(i) **Was an earlier form of the manuscript uploaded to a preprint server (e.g. medRxiv)? If 'Yes', please give a link or DOI**

DOI

✓ I acknowledge permission of the publisher named under 1d to include in this thesis portions of the publication named as included in 1c.

2. For a research manuscript prepared for publication but that has not yet been published (if already published, please skip to section 3):

(a) **What is the current title of the manuscript?**

(b) **Has the manuscript been uploaded to a preprint server (e.g. medRxiv)?**
If 'Yes', please give a link or DOI:

(c) **Where is the work intended to be published?**

(d) **List the manuscript's authors in the intended authorship order:**

(e) **Stage of publication:**

3. For multi-authored work, please give a statement of contribution covering all authors (if single-author, please skip to section 4):

Clementine, Devon, Sebastian,and Stefano all contributed equally to developing and analyzing the student-teacher setup used to study the impact of initialization, which led to the paper. In the paper, Clementine also led the disentanglement experiments. Sebastian focused on the student-teacher framework and its role in understanding continual learning, as well as conducting the related experiments. Devon led the linear network analysis. Andrew M. Saxe and Stefano Sarao Mannelli provided supervision throughout the project. All authors contributed to writing the appendix and refining the manuscript.

4. In which chapter(s) of your thesis can this material be found?

Chapter 2 and 4

e-Signatures confirming that the information above is accurate

This form should be co-signed by the supervisor/senior author unless this is not appropriate (e.g. if the paper was a single-author work):

Candidate: Clementine Domine	**Date:** 14/ 04/2025
Supervisor/Senior Author: Andrew Saxe	**Date:** May 8 2025

UCL Research Paper Declaration Form: Referencing the Doctoral Candidate's Own Published Work(s)

1. For a research manuscript that has already been published (if not yet published, please skip to section 2):

(a) **What is the title of the manuscript?**

(b) **Please include a link to or DOI for the work:**

(c) **Where was the work published?**

(d) **Who published the work?**

(e) **When was the work published?**

(f) **List the manuscript's authors in the order they appear on the publication:**

(g) **Was the work peer reviewed?**

(h) **Have you retained the copyright?**

(i) **Was an earlier form of the manuscript uploaded to a preprint server (e.g. medRxiv)? If 'Yes', please give a link or DOI**

If 'No', please seek permission from the relevant publisher and check the box below:

✓ I acknowledge permission of the publisher named under 1d to include in this thesis portions of the publication named as included in 1c.

2. For a research manuscript prepared for publication but that has not yet been published (if already published, please skip to section 3):

(a) **What is the current title of the manuscript?**
Position: Solve Layerwise Linear Models First to Understand Neural Dynamical Phenomena (Neural Collapse, Emergence, Lazy/Rich Regime, and Grokking)

(b) **Has the manuscript been uploaded to a preprint server (e.g. medRxiv)?**
If 'Yes', please give a link or DOI: DOI

(c) **Where is the work intended to be published?**
International Conference on Machine Learning (ICML) Position Paper track 2025

(d) **List the manuscript's authors in the intended authorship order:**
Yoonsoo Nam, Seok Hyeong Lee, Clementine Domine, Yea Chan Park, Charles London, Wonyl Choi, Niclas Goring, Seungjai Lee

(e) **Stage of publication:**
Under review, (Accepted)

3. For multi-authored work, please give a statement of contribution covering all authors (if single-author, please skip to section 4):

Yoonsoo wrote the paper and produced most figures including experiments in appendix G. Seokhyeong solved, checked, and discussed all the mathematics in the paper including the main technical contribution: Theorem 1. Clementine wrote Section 6 and Appendix F (including all the figures/experiments) on imbalanced networks and how they relate to grokking. She also discussed the main text and literature reviewed linear neural networks and neuroscience. Yeachan wrote a literature review on grokking in Appendix G and discussed grokking in Section 7. Charles checked the clarity and overall structure of the paper from the view of computer science. Wonyl discussed the overall structure, mainly focusing on the introduction and setup, of the paper from the view of physicists. Niclas helped initiate the project and specified the unmet needs. Seungai Lee was the corresponding author, managing the project and providing the view of mathematics.

4. In which chapter(s) of your thesis can this material be found?

Chapter 3 and Chapter 5

e-Signatures confirming that the information above is accurate

This form should be co-signed by the supervisor/senior author unless this is not appropriate (e.g. if the paper was a single-author work):

Candidate: Clementine Domine	**Date:** 14/ 04/2025
Supervisor/Senior Author: Andrew Saxe	**Date:** May 8 2025

UCL Research Paper Declaration Form: Referencing the Doctoral Candidate's Own Published Work(s)

1. For a research manuscript that has already been published (if not yet published, please skip to section 2):

(a) **What is the title of the manuscript?**

(b) **Please include a link to or DOI for the work:**

(c) **Where was the work published?**

(d) **Who published the work?**

(e) **When was the work published?**

(f) **List the manuscript's authors in the order they appear on the publication:**

(g) **Was the work peer reviewed?**

(h) **Have you retained the copyright?**

(i) **Was an earlier form of the manuscript uploaded to a preprint server (e.g. medRxiv)? If 'Yes', please give a link or DOI**

If 'No', please seek permission from the relevant publisher and check the box below:

✓ I acknowledge permission of the publisher named under 1d to include in this thesis portions of the publication named as included in 1c.

2. For a research manuscript prepared for publication but that has not yet been published (if already published, please skip to section 3):

(a) **What is the current title of the manuscript?**
Learning dynamics in linear recurrent neural networks

(b) **Has the manuscript been uploaded to a preprint server (e.g. medRxiv)?**
No

(c) **Where is the work intended to be published?**
International Conference on Machine Learning (ICML) Main Comference Paper track 2025

(d) **List the manuscript's authors in the intended authorship order:**
Alexandra Maria Proca, Clémentine Carla Juliette Dominé, Murray Shanahan, Pedro A. M. Mediano

(e) **Stage of publication:** Under review (Accepted).

3. For multi-authored work, please give a statement of contribution covering all authors (if single-author, please skip to section 4):

Alex developed the framework and worked on the applications. Clementine C. J. Domine and Alex collaborated on the application to the ”Rich and Lazy” section of the paper. Murray Shanahan and Pedro A. M. Mediano provided supervision throughout the project. All authors contributed to writing the appendix and refining the manuscript.

4. In which chapter(s) of your thesis can this material be found?

Chapter 2 and 3

e-Signatures confirming that the information above is accurate

This form should be co-signed by the supervisor/senior author unless this is not appropriate (e.g. if the paper was a single-author work):

Candidate: Clementine Domine	**Date:** 14/ 04/2025
Supervisor/Senior Author: Andrew Saxe	**Date:** May 8 2025

Acknowledgements

I feel deeply grateful, lucky, and privileged to have had the opportunity to pursue this PhD and work in science. This was only made possible by the unwavering support of those around me—friends, family, and the inspiring teachers who encouraged my growth. To everyone who has been a part of this journey in big and small ways—thank you.

To My Mentors and Supervisors

Thank you to all the mentors and supervisors who have guided me over the years. To *Andrew*, thank you for taking me on as one of your first students at Gatsby. Your trust, patience, and flexibility gave me the freedom to grow into the scientist I am today. Thank you for always sharing your excitement for science; it has been a challenge and an honour working with you. To *Caswell*, an extraordinary researcher and mentor, thank you first for recommending me to apply to Gatsby. Your belief in me, availability, and uplifting guidance have meant so much throughout this journey. To *Kim Stachenfeld*—thank you for your time, encouragement, and deep understanding. And to other mentors I've been fortunate to learn from and been inspired by: *Emmanuelle Rodolà*, *James Fitzgerald*, *Pedro Mediano*, *Petar Veličković*, *Marcus Stephenson-Jones*, *Nina Miolane*, *Robin Bruinsma*, and *William Bertsche*—thank you.

To My Co-authors and Colleagues

I believe science doesn't happen in isolation—it thrives through collaboration. Discovery is a shared effort shaped by open dialogue, diverse perspectives, and the support of a community that challenges and uplifts one another. Every insight in this thesis has been shaped, directly or indirectly, by conversations, collaborations, and the generosity of others who offered their time, expertise, and encouragement. I'm grateful to have been part of such a collaborative journey.

To my co-authors: *Rodrigo Carrasco Davis*, possibly the wisest person I know—your words, "If you work every day from 9 to 5, something will come out of it," carried me and kept me sane through this PhD. Thank you for your kindness and friendship. *Devon Jarvis*, a constant reminder of what really matters with an intact excitement for science—thank you for your generosity, grounded perspective, and friendship. *Sébastian Lee*, a really great friend and a humble scientist, is an even better co-pilot for road trips, with endless cool stories. *Marco Pegoraro*, thank you for your patience in welcoming me into a field I knew nothing about... and for the ice creams. *Lukas Braun*, thank you for sharing the journey of writing my very first paper. *Nicolas Anguita*, thank you for bringing fresh energy, excitement, and a bright vision to my research. *Daniel Kunin*, thank you for the opportunity to work together. It's been a challenging journey—from late-night video calls to the rewarding work, but most of all, I've cherished the outdoor adventures. *Will Dorrell*, for your endless energy and wildly curious and open mind. *Yoonsoo Nam*, for being a quiet reminder to stay kind, always. *Alexandra Proca*—a skilful, absolutely brilliant researcher, it has been a pleasure working with you. *Luke Hollingsworth*—now a great friend, but always my "first Master's student"—thank you. And to *Tom George*, *Andrea Deac*, *Emmette Thompson*, *Lars Rollik*, *Benjamin Waked*, *Georgia Mills*, *Jasvin Kaur*, *Allan Raventós*, *Feng Chen*, *Ben Geva*, *Niko Sirmpilatze*, *Adam Tyson*, and *David Klindt*—thank you for your trust, your brilliant ideas, and all the scientific fun we've had along the way. A special thank you to *Erin* and *Stefano* for your constant support and guidance—it's meant more than I can say.

To my colleagues and friends in the greater Gatsby community who shared the PhD journey with me—especially the *BOE*—thank you for the laughter, the moral support that kept me going, and the unbelievably great research. A special shout-out to *Jin*, *Yedi*, *Nishi*, *Kira*, *Dimitri*, *Aditya*, *Vasco*, *Valentina* and everyone else I had the joy of sharing this journey with—thank you. To the *Saxe Lab*—grateful for the friendships, and thank you for making science fun, welcoming, and a place where I truly felt like I belonged. To the *Gladia Lab* in Rome—thank you for the warm welcome and for making me feel at home, so much so that I came back multiple times. To the *Columbia community* in NY, and especially the *Stackenfeld Lab*—thank you for the thoughtful conversations and warm welcome. A special thanks to *Samuel Lippl*, a researcher of incredible depth—thank you for the encouragement, insights, and friendship. To the *Gatsby* staff—*Barry*, *Mike*, *Despina*, and *I-Chun*—thank you for all the tireless behind-the-scenes support.

To My Friends

To the *Camden girls*—you have been my rock, my refuge, and the sisters I never knew I needed. I am so proud to be surrounded by such strong, independent, inspiring women. You supported me through my PhD, helped me make important decisions, and helped me step outside the academic bubble. I will carry your strength with me wherever I go. To *Agathe*, *Anaïs*, *Maddie*, *Solenne*, *Emma*, *Adey*, *Andrea*, *Tanaya*, *Rocio*, *Anita*, *Camille*, *Sarah*, *Esther*, *Alice*, *Devon*, and *Will*—thank you, from the bottom of my heart. To my *University of Manchester* friends—the Gongers: *Sarouch*, *Mariella*, *Pilou*, *Clara*, *Alicia*, *Eddie*, *Luismi*, *Chris*, *Winnie*, *Dario*, *Elena*, *Camille*, and *Anna*—probably the most chaotic, supportive, loving, and fun group of friends I have, from all corners of the world. It's been such a joy not only sharing our university experiences but also growing together as we dive into real adult life. I'm so proud of each of your journeys and beyond grateful to call you my friends. To the friends I don't see as often as I'd like, yet who have always been there—keeping our bond strong across the distance and filling every reunion with joy—*Chloé*, *Nadège*, *Chloé*, *Maria*, *Elo*, *the Lamphoutacoul*, and *the Bintein*—thank you for the laughter and the memories. To my childhood friends—*Johana*, *Martin*, *Morgan*, and *Alisée*—thank you for your enduring presence and the constancy you've brought to my life.

To My Family

I am grateful to my family for their unconditional love, understanding, and support, especially when sacrifices had to be made for me to pursue my passions and dreams. I am so thankful to have you in my life. To my parents—thank you for sending me to study abroad, even when Mom was still nervous about me crossing the road alone not long before that. Your courage became my own. *Vous avez dû voir plein de jolis petits poissons, avec tout ce temps passé avec un tuba. Merci! :)*

To *Grand-Mère*—you are a role model, an inspiration, and a brilliant woman in science. To *Mame* and *Pape*, and to the *Labrune* and *Varnieux* and *Dominé* families—thank you for your love and for all the little things that helped shape who I am. To my brother *Aurelien*—thank you for always being supportive, caring, and acting as a great motivator. Competing with you (even if it was mostly in my head) constantly pushed me to do better. And to *Alex*, my twin sister and my other half—thank you for always being by my side and for allowing me to chase my dreams in my own way. Side by side or miles apart, twins will always be connected by heart. To *Milo*, a bright, curious, and caring person—thanks.

To Marco

Finally, I would like to thank *Marco Fumero* for being part of my PhD journey. Your time, guidance, and encouragement have given me the opportunity to reach my full potential. Beyond the professional setting, you are my sunshine; you are my today and all my tomorrows. Thank you for being by my side, unwavering, always reminding me of what matters most. Thank you for being careful and making every day into a beautiful adventure. *Je t'aime*.

Contents

Chapter 1

Introduction

The more you know, the more you realise you don't know.

– Aristotle

A defining characteristic of human learning is our remarkable sensitivity to prior knowledge—what we already know profoundly influences how we acquire new information [44]. For example, once we have learned the attributes of several animals, recognising and categorising a new one becomes significantly more efficient [192, 191, 89]. In contrast, an infant without prior exposure to different animals will exhibit distinct learning dynamics, forming representations that differ fundamentally from those of an experienced learner. These differences in learning dynamics highlight the role of prior knowledge in shaping how information is processed and integrated.

In machine learning, prior knowledge similarly influences learning across a range of paradigms, including reversal learning [84], transfer learning [276, 273, 161, 101], curriculum learning [30], and meta-learning [136]. While structured priors can accelerate learning and improve generalisation, they can also introduce constraints that hinder adaptation, as seen in the phenomenon of catastrophic interference in continual learning [212, 148, 306].

Despite the well-established importance of prior knowledge, fundamental questions remain about how structured initial conditions shape the inductive biases of artificial and biological neural networks, how they influence learning dynamics, and what types of representations they support [31, 186]. Understanding how initialisation impacts gradient-based learning could lead to improved pretraining strategies and offer insights into challenges such as catastrophic forgetting in continual learning.

Here, we present frameworks for systematically exploring these effects in a controlled setting. Our focus is on a fundamental form of prior knowledge in deep networks: the initial network state and the representations learned from previous tasks. Prior work has shown that certain initialisations place networks in a 'lazy' regime, where internal representations do not change through learning, while other initialisations place networks in a 'rich/feature learning' regime where internal representations evolve to fit a task [51, 130].

We investigate the learning regime dynamics of neural models across different architectures, datasets, and activation functions in simplified toy models to better understand the role of prior knowledge. Each of these factors plays a crucial role in shaping learning trajectories, influencing a model's ability to extract meaningful task-specific features [243, 215, 22]. Our analysis [243, 69, 158, 39, 67] moves beyond previous approaches that depend on restrictive assumptions, which often limits their applicability to real-world scenarios. Ultimately, we investigate how neural representations emerge throughout training, with the goal of developing a theoretical framework for their formation in the context of neuroscience.

1.1 Thesis structure

This thesis opens with a background chapter that introduces the broader field of machine learning, emphasising its connections to cognitive science and neuroscience. A concise literature review then follows, concisely presenting key research that shaped this work. The main body is divided into two key sections: The first investigates the factors driving rich and lazy learning regimes by analysing exact learning dynamics in deep linear networks with prior knowledge, offering both theoretical insights and practical implications. The first section examines the factors driving rich and lazy learning regimes, focusing on exact learning dynamics in deep linear networks with prior knowledge and providing insights into their theoretical foundations as well as practical application. The second section extends the analysis beyond linear networks, exploring nonlinear ReLU networks and practical architectures. The thesis concludes with a general discussion, summarising key findings and proposing directions for future research.

1.2 Contributions

The contributions of this thesis are threefold:

1. We advance theoretical understanding by defining the spectrum from the lazy to the rich regime, providing a clearer framework for characterising different learning dynamics.

2. We extend deep learning theory by exploring a broader range of initialisation schemes and settings that yield exact learning dynamics.

3. We demonstrate that theories derived from toy linear networks and non-linear networks have practical implications for real-world learning scenarios relevant to neuroscience and machine learning in practice.

Chapter 2

Background

Standing on the shoulders of giants.

– Isaac Newton

2.1 Background

This research lies at the intersection of machine learning and neural network theory, focusing on foundational aspects of deep learning. We begin by introducing the core concepts of deep learning, followed by a concise historical overview of deep artificial neural networks, highlighting the gaps that motivate this study. We then examine the links to cognitive science and neuroscience, where deep learning offers a powerful framework for exploring key questions—particularly those related to learning mechanisms and the formation of representations shaped by prior knowledge in biological systems.

2.1.1 Deep neural networks

Deep neural networks consist of multiple layers of interconnected neurons, each optimised to process data, mimic cognitive functions, and develop abstractions. The network starts with an input layer that receives vectors of dimension N_i. These inputs are passed through a linear transformation, parametrised by the weight matrix $\mathbf{W}_1 \in \mathbb{R}^{N_h \times N_i}$, followed by a nonlinear activation function—together constituting the first hidden layer. Each subsequent hidden layer applies a similar transformation, using weight matrices typically shaped $\mathbb{R}^{N_h \times N_h}$, again followed by nonlinearities. Finally, the output layer uses the weight matrix $\mathbf{W}_L \in \mathbb{R}^{N_o \times N_h}$ to generate the model's predictions. Crucially, a layer encompasses both the linear transformation and the activation function.

Consider a supervised learning task where input vectors $\mathbf{x}_n \in \mathbb{R}^{N_i}$, drawn from a set of P training pairs $\{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^{P}$, must be mapped to their corresponding target output vectors $\mathbf{y}_n \in \mathbb{R}^{N_o}$. An L-layer network computes the output as:

$$\hat{\mathbf{y}} = f(\mathbf{W}_{L-1} f(\mathbf{W}_{L-2} \cdots f(\mathbf{W}_2 f(\mathbf{W}_1 \mathbf{x})) \cdots)) \tag{2.1}$$

where the function $f(\cdot)$, applied element-wise, introduces a neural nonlinearity. Common activation functions include ReLU, sigmoid, and tanh.

We train the network by solving the following optimisation problem:

$$\min_{\mathbf{W}_{L-1}, \ldots, \mathbf{W}_1} \sum_{n=1}^{P} \mathcal{E}(\mathbf{y}_n, \hat{\mathbf{y}}_n) \tag{2.2}$$

where $\mathcal{E}(\cdot,\cdot)$ is an error function that penalises discrepancies between the desired output $\mathbf{y}_n$ and the predicted output $\hat{\mathbf{y}}_n$.

Typically, this objective is approximately minimised through backpropagation, by iteratively adjusting each weight matrix in the direction of the negative gradient of task performance:

$$\Delta W_l = -\eta \sum_{n=1}^{P} \frac{\partial \mathcal{E}(\mathbf{y}_n, \hat{\mathbf{y}}_n)}{\partial W_l}, \quad l = 1, \ldots, L-1, \tag{2.3}$$

where η is a small learning rate parameter. This update rule makes small adjustments to the weights, aiming to improve task performance most effectively. However, as will be discussed, the successful implementation of Eqn. 2.3 in practice has taken decades of development.

2.1.2 A short history of deep learning

The origins of deep artificial neural networks (ANNs) can be traced back to the mid-20th century, drawing inspiration from biological brain processes [263, 194, 248]. However, their full potential was not realised until the 2000s. Early neural models, including perceptrons and the first multilayer networks, showed initial promise but were hampered by significant limitations [262]. One of the most critical challenges arose during optimisation using backpropagation [142] In deep networks, however, gradients often diminish as they are propagated through successive layers, effectively stalling learning. Compounding this issue were the constraints of limited computational power, the absence of large-scale datasets, and prevailing scepticism within the scientific community. These factors led to a prolonged period of stagnation in Artificial Intelligence (AI) research—commonly referred to as the ”AI Winter”—during which both funding and interest in neural networks waned, delaying further advancements until pivotal breakthroughs revived the field.

The field has witnessed an exponential acceleration, particularly throughout the early 2000s, driven by several key factors. One of the primary drivers has been the continuous increase in computational power, both in terms of the scale of available computing resources and advancements in hardware technologies [222, 10]. Additionally, significant progress in neural network architectures [163, 209, 286, 109, 147], the development of more effective activation functions [5, 77], and advances in optimisation algorithms [146, 235] have played crucial roles in shaping modern AI systems. Additionally, the scale and quality of datasets available for training have expanded significantly, further enhancing the capabilities of AI models [115, 268]. Further breakthroughs in network initialisation techniques [164, 105, 243, 118, 223] have facilitated the training of deeper networks by addressing challenges such as vanishing and exploding gradients, pointing to the importance of prior knowledge in these networks. Collectively, these advancements have accelerated the evolution of deep learning, leading to increasingly powerful and efficient AI systems. Deep learning has now revolutionised machine learning, showcasing exceptional capabilities across various tasks, including image recognition and natural language processing.

However, our limited ability to interpret deep learning models—frequently described as opaque ’black boxes’—continues to hinder both our confidence in their decision-making and their effective deployment in real-world applications. The theoretical understanding of how information is represented in these models — which could potentially explain their success — remains largely elusive. One particularly concerning consequence of this issue is the presence of biases in models [195, 108], which can lead to unfair and potentially harmful decision-making [2]. As a result, research areas like mechanistic interpretability [257, 79, 204], along with deep learning theory and toy models [236, 243], have emerged—seeking to understand what and how neural networks learn - often by making simplifying assumptions. In particular, these studies highlight that several factors shape the representations learned by neural networks and, consequently, the biases they develop [50]. These factors include model architecture, activation functions, optimisation techniques, initialisation strategies, and data selection [106, 237]. A deeper understanding of how these factors influence learning in neural networks is critical for advancing the reliability, transparency, and fairness of deep learning systems. Furthermore, this understanding could lead to improved performance and offer insights into challenges such as catastrophic forgetting in continual learning.

As deep learning becomes increasingly embedded in real-world applications, addressing these challenges is essential. Understanding different learning paradigms—including how, what, and when models learn—and their inherent biases is crucial, with important implications for machine learning in practice and neuroscience.

2.1.3 Deep learning and the brain

The deep learning revolution has not only transformed computer science and its applications but has also gained traction as a model for brain function, a field often referred to as NeuroAI [229, 241]. Much like artificial intelligence, the brain can be considered a black box, with many of its inner workings still unknown. As a result, both theoretical neuroscience and deep learning theory share a common goal: to describe and understand the representations and behaviour of neural networks. While deep learning models have been criticised for their lack of biological plausibility—particularly regarding the implementation of the backpropagation algorithm [292, 56]—a growing body of research is actively exploring the parallels between biological and artificial neural networks (ANNs) representations [89, 309, 224, 299, 229, 66]. Several review articles have synthesised these insights [259, 141, 150].

One key research direction explores why neural networks, both biological and artificial, develop similar representations when trained on similar tasks [116, 159, 198, 149, 200, 95, 144, 151] These similarities emerge across multiple levels of analysis—not only at the level of individual neurons but also in the structured population-level representations found within and across brain regions. Below, we discuss some of the main results of research on representation learning and the brain.

Extensive research on the visual system [153, 249, 151, 179, 298] has demonstrated striking parallels between biological models and deep learning architectures such as transformer, convolutional neural networks (CNNs) and Residual Neural Networks (ResNets). Interestingly, at the neuronal representation level, these networks develop features resembling Gabor-like filters, which closely mirror receptive fields observed in early visual processing in biological systems [305, 85]. Furthermore, networks designed to mimic representations observed in the brain—such as constraining the filters in the first layer to follow a Gabor function— have been shown to outperform standard convolutional networks [7]. Altogether, Gabor filters have emerged as important representational features in image recognition in both biological and artificial vision systems. They offer a successful mathematical model of receptive fields, capturing spatial frequency, orientation selectivity, and phase sensitivity—key properties for edge detection and texture analysis.

Beyond visual processing, deep networks have demonstrated the ability to replicate neural representations involved in spatial navigation. Research using deep and recurrent neural networks (RNNs) trained on navigation tasks has revealed emergent activity patterns strikingly similar to grid cells [264, 24, 293]—a class of spatially modulated neurons located in the medial entorhinal cortex that fire in a characteristic hexagonal grid pattern, crucial for encoding spatial location and supporting memory functions [114]. Remarkably, such representations have also been observed across species, suggesting they may reflect a fundamental mechanism for path integration in biological and artificial learning systems [114, 65].

Taken together, these models effectively reproduce key representational properties observed in biological systems, underscoring deep learning's potential to capture fundamental neural mechanisms [95]. However, the similarity in representation does not necessarily suggest that the two learning systems are governed by the same underlying principles, raising concerns about whether we are truly capturing the brain's behaviour. For instance, different models can achieve similar representations using distinct architectures, initialisation and learning algorithms, leading to considerable degeneracy in both the models and their parameters [220, 299]. This is well exemplified in the Brain-Score framework, which shows that different models often perform similarly overall, with some variability across brain regions or evaluation metrics [250]. This overlap makes it challenging to distinguish between models or determine which best represents biological computation.

An expanding body of research, including the present study, emphasises the importance of adopting a temporal perspective to illuminate the emergence and development of neural representations [243, 241, 87]. Rather than treating representations as static endpoints, this approach examines how they evolve throughout the learning process. This temporal framing draws on insights from artificial neural networks, where the dynamics of learning—and notably, the contrast between rich and lazy regimes—play a fundamental role in shaping the geometry and functional properties of representations.

Previous work has established conceptual parallels between these computational regimes and patterns of neural representation observed in the brain, suggesting that rich and lazy learning provide a useful framework for interpreting task-specific and task-agnostic coding strategies [87]. Strikingly, both artificial and biological neural systems have been shown to support a wide spectrum of representations that vary with task structure, brain region, and prior experience [207, 96, 87, 210, 282]. However, a comprehensive theoretical account of how such regimes arise, how they interact, and what determines their functional roles remains an open challenge in the field.

Artificial neural networks offer a tractable and flexible model system for addressing these questions. Their learning dynamics can be finely controlled by manipulating architectural features, initialisation schemes, task structures, and input distributions. This experimental precision enables systematic exploration of the factors that influence learning trajectories and, ultimately, the formation of diverse representations.

The parallels between deep learning and neuroscience are becoming increasingly compelling with the emergence of architectures that not only surpass human performance but also mimic cognitive processes in various tasks [251, 229, 160]. Investigating the emergence of similar representations in both biological and artificial systems may reveal shared underlying principles, while understanding their differences can shed light on how distinct learning mechanisms shape neural computation.

2.2 Related works

In this section, we review the literature most relevant to this thesis. We begin by examining previous works on rich and lazy learning, followed by an analysis of the learning settings, architecture and analytical frameworks relevant to this thesis. We identify the gaps in the existing literature this research aims to address.

2.2.1 Rich and Lazy regime

Prior research has demonstrated that different network initialisations give rise to distinct learning regimes: a ”lazy” regime, in which internal representations remain largely unchanged throughout training, and a ”rich” or ”feature-learning” regime, where representations evolve to better fit the task [130, 51]. We review below the factors that drive the emergence of these learning regimes and characterise the representations they produce in neural networks. Furthermore, we discuss their parallels with findings in neuroscience, where a similar dichotomy between non-linear mix-selectivity and manifold learning has been observed.

2.2.1.1 Rich and Lazy regimes in neural networks

The *lazy* regime follows from a fundamental phenomenon in overparameterised neural networks: during training, in the limit of infinite width, these networks can fit the training data with a variation of their parameters which is much smaller than their initialisation scale. [51]. As a result, their learning dynamics resemble those of kernel regression, governed by the Neural Tangent Kernel (NTK) that remains fixed during training, and demonstrate exponential learning dynamics across various architectures, including feedforward and recurrent networks [75, 130, 73, 8, 9, 313, 81]. This phenomenon, referred to as the lazy or kernel regime, typically emerges in the infinite-width limit of neural networks and can be induced by initialising the weights with sufficiently large variance [130, 51]. Under these conditions, the network exhibits approximately linear behaviour in its parameters and remains close to its initialisation throughout training. While the *lazy* regime offers valuable insights into how networks converge to a global minimum, it does not fully account for the generalisation capabilities of neural networks trained with standard initialisations

-such as Xavier or He initialisation [105, 118]. It is,therefore, widely believed that another regime, driven by small or vanishing initialisations, underpins some of the successes of neural networks.

In contrast, the *rich* feature-learning regime is characterised by an NTK that evolves throughout training, accompanied by non-convex dynamics that navigate saddle points [23, 243, 247, 131]. This regime features sigmoidal learning curves, feature adaptation to the data structure, and simplicity biases, such as low-rankness [170] sparsity, [294] or specialisation [106]. Numerous studies typically consider the rich regime which emerges at small initialisation [51, 99]. However, even at small initialisation scales, differences in weight magnitudes between layers can induce the *lazy* learning regime [21] – highlighting the significance of both *absolute scale* (initialisation variance) and *relative scale* (difference in weight magnitude between layers) in generating diverse learning dynamics [18]. Aside from the works of Liu et al. [176], Emami et al. [81], most theoretical investigations into rich and lazy learning have focused on feedforward networks.

The precise characterisation of rich learning—and the features it acquires—often depends on the particular task under consideration, and is usually defined in contrast to lazy learning. A comprehensive understanding of the continuum between rich and lazy regimes, including their dynamics across diverse architectures, remains an open challenge in the field. This thesis seeks to address this gap by offering a clearer characterisation of these regimes through the use of simplified toy models, highlighting the crucial role played by relative scaling at initialisation.

2.2.1.2 Rich and lazy regimes in the brain

The distinction between *rich* and *lazy* learning regimes may offer valuable insights for neuroscience, where neural representations are frequently characterised as either task-specific or task-agnostic, depending on the experimental context [87, 210, 282].

This analogy is conceptually compelling: when neural representations adapt in a task-specific way, they mirror the behaviour of the *rich* learning regime; conversely, representations that exhibit minimal change and remain stable across tasks reflect the *lazy* regime and are inherently task-agnostic.

However, a key limitation of current experimental work is that neural representations are typically measured only at the end of learning. As a result, these snapshots provide limited insight into the dynamics of how representations evolve throughout the learning process. Consequently, the same final representation could, in principle, emerge from either rich or lazy learning dynamics. To address this ambiguity, future research should aim to characterise the temporal evolution of neural representations during learning. Understanding these dynamics is essential for distinguishing between learning regimes and for interpreting the functional role of representational changes—or lack thereof—across brain regions and tasks.

Nonetheless, certain types of neural representations have been linked to properties shaped by the underlying learning regime. In particular, the *lazy* learning regime has been associated with random non-linear mixed selectivity, whereby task-relevant variables are projected into a high-dimensional space through random feature mixing [226, 271, 230, 33]. These randomly mixed-selective representations are well suited to generalisation across tasks due to their high dimensionality: by embedding input randomly and non-linearly into a sufficiently large feature space, task-relevant distinctions often become linearly separable. As a result, simple linear readouts may be sufficient for solving complex tasks. This suggests that effective task learning can occur rapidly, without necessitating substantial changes to the underlying representations —indicative of the *lazy* regime. By contrast, the *rich* regime involves structured representational changes and has been associated with linear mixed selectivity [282, 206] and manifold learning. In this regime, the brain encodes information on low-dimensional, task-specific manifolds—examples of which include the grid-like codes observed in entorhinal cortex cells [47, 33, 90].

The task-specific nature of these representations is often interpreted as evidence of alignment with task-relevant vectors. For example, Flesch et al. [90] showed that neural population codes observed through human fMRI reflect representational geometries that align with those produced by the rich regime in a model network. Altogether, task-specific representations are often taken as evidence of alignment with the *rich* regime, while the absence of task-specific characteristics may suggest a *lazy* regime. However, the broader conditions under which such structured representations emerge—across different brain regions, task demands, prior knowledge, and task structures—remain largely unexplored.

The computational models explored in this thesis — based on neuroscience-relevant architectures such as deep feedforward and recurrent neural networks — provide a robust framework for formulating new theories about how neural representations arise and develop in the brain [156]. These models enable systematic manipulation of key variables, allowing researchers to isolate and examine the factors that shape different learning regimes, including task demands, environmental structure, architectural design, and the influence of prior knowledge. Moreover, they provide a unique opportunity to investigate the temporal dynamics of learning, potentially leading to novel theoretical insights and predictions. Significant gaps remain in our understanding of the dynamic of neural representations and the emergence and progression of complex cognitive functions across the lifespan [241]. By addressing this gap, we may foster experimental exploration of this pressing open question in the field.

2.2.2 Learning setting

Prior knowledge plays a crucial role in shaping both the process and outcome of learning. We outline below different learning setting in which prior knowledge impacts the learning dynamics.

2.2.2.1 Continual Learning

Continual learning refers to the ability of an agent to learn from a sequence of tasks without forgetting previously acquired knowledge. In a continual learning regime, a biological agent can seamlessly switch between tasks and retain prior knowledge while learning new tasks. However, this is a major challenge for artificial agents, as they often struggle with catastrophic forgetting. When trained on a new task, artificial models tend to overwrite the knowledge learned from previous tasks, making it difficult to retain information from earlier stages. This phenomenon can also negatively impact other areas, such as reinforcement learning, multi-task learning, and lifelong learning [11, 278, 261]. Overcoming this pathology may unlock the potential for versatile and flexible learning algorithms. Recent efforts to mitigate catastrophic forgetting [212, 60] have led to the development of regularisation strategies that promote specialisation, such as elastic weight consolidation [148] and synaptic intelligence [306]. Finally, prioritised replay has been proposed as a mechanism to combat catastrophic forgetting. Replay methods include storing data from earlier tasks to interleave when learning new tasks [192]. In Chapters 3 and 4, we investigate how initialisation impacts the dynamics of continual learning and evaluate the effectiveness of various methods for mitigating forgetting under different initialisation schemes.

2.2.2.2 Reversal Learning

The reversal framework, akin to the continual learning paradigm discussed earlier, represents a form of sequential learning. Specifically, Reversal learning is the process where an individual changes their behaviour by unlearning an old habit and forming a new one in response to changing situations. This process requires inhibiting a formerly rewarded response and shifting to an alternative, making it a key measure of cognitive flexibility in both humans and animals [129]. Cognitive flexibility in animals during reversal learning displays considerable variability, with behaviours ranging from rapid switching to gradual adaptation [133]. Similarly, in deep neural networks, we observe a diverse range of reversal learning profiles. Notably, deep linear networks can exhibit both catastrophic slowing and rapid reversal [168]. In

Chapter 3, we present a theoretical framework that may underpins these observations.

2.2.2.3 Transfer Learning

Transfer learning involves the transfer of knowledge across different domains [310, 279]. Its origins can be traced back to educational psychology, which suggests that knowledge transfer occurs through the generalisation of prior experiences. A practical example is that a person who has learned to play the guitar may acquire trumpet-playing skills more quickly than someone without prior musical training, as both instruments share fundamental musical concepts. The transfer learning framework shares similarities with the continual learning paradigm discussed earlier, particularly due to its sequential learning approach. Importantly, continual learning also focuses on minimising forgetting. Previous research has uncovered a fascinating interplay between transfer and forgetting [168, 167, 17, 64], as well as its connection to the learning regime the network operates in [121]. For example, lazy representations tend to perform poorly at transfer, whereas rich representations support greater transfer. In a similar setup, the benefits of leveraging features learned from upstream tasks for new tasks were examined in single-layer and two-layer networks [63, 101, 270]. These studies explored how task similarity and data scarcity impact downstream task performance [167, 168]. In Chapter 3, we investigate how initialisation impacts transfer learning.

2.2.2.4 Fine-Tuning

Fine-tuning, a common technique for transfer learning, pre-trains neural networks on a larger auxiliary task before training on a downstream task of interest. Rather than training a model from scratch, fine-tuning enables the transfer of knowledge from a broad, general dataset to a more specialised domain. This technique is especially valuable in deep learning, where models trained on extensive datasets, such as ImageNet for computer vision or large text corpora for natural language processing, can be effectively adapted to new tasks with minimal labelled data [213, 183]. As the computational costs of training large models continue to rise, fine-tuning has emerged as a more efficient alternative. However, despite its widespread use, the underlying dynamics and effects of fine-tuning remain poorly understood. Previous research has advanced in this area by shedding light on the implicit regularisation effects that arise from pre-training followed by fine-tuning, particularly in diagonal linear networks [173].

In Chapter 3, we explore the fine-tuning dynamics of two-layer linear networks. Although structurally simple, these models play a foundational role in fine-tuning strategies, particularly through their use in Low-Rank Adapters (LoRA) [126]—a widely employed technique aimed at reducing computational demands. The LoRA parameterisation, given by $\Delta\mathbf{W} = \mathbf{A}\mathbf{B}$, effectively embeds a two-layer linear sub-network within the larger language model architecture. Recent research has investigated how factors such as the initialisation of the low-rank matrices $\mathbf{A}$ and $\mathbf{B}$, as well as architectural properties like infinite width, influence fine-tuning performance—frequently linking these effects to the underlying learning regime[183, 117, 62]. Notably, low-rank fine-tuning has been shown to often operate within a so-called *lazy* or *kernel* regime [183]. Specifically, the standard fine-tuning and its LoRA-based variant were shown to perform comparably to their respective kernel approximations. This regime is characterised by minimal deviation from the pre-trained parameters and strong retention of the model's original representations.

2.2.3 Architectures

Neuroscience studies have shown that different regions of the brain possess unique structural architectures, with variations in their connectomics—the mapping of neural connections that underlie brain function—potentially reflecting their specialised functional roles [162]. Similarly, in machine learning, structural parameters such as depth, width, and connectivity are systematically analysed to assess their influence on learning dynamics, structured representations, and overall model performance [67, 158, 37]. Architectural design is thus a crucial factor in the learning process for both biological and artificial neural networks. In the following sections, we review key deep learning architectures that are particularly relevant to this thesis, emphasising their success in enhancing our understanding of neural representations and computational principles.

2.2.3.1 Linear networks

Our work builds upon a rich body of research on deep linear networks, which, despite their simplicity, have proven to be valuable models for understanding more complex neural networks [23, 93, 243]. Previous research has extensively analyzed convergence [12, 72], generalisation properties [161, 217, 125], and the implicit bias of gradient descent [14, 294, 50, 157] in linear networks.

These studies have also revealed that deep linear networks have intricate fixed-point structures and nonlinear learning dynamics in parameter and function space, reminiscent of phenomena observed in nonlinear networks [13, 161]. Seminal work by Saxe et al. [243] laid the groundwork by providing exact solutions to gradient flow dynamics under task-aligned initialisations, demonstrating that the largest singular values are learned first during training. This analysis has been extended to deep linear networks [13, 14, 312] with more flexible initialisation schemes [103, 272, 104].

In Chapter 3, we directly builds on the matrix Riccati formulation proposed by Fukumizu [93], which extends these solutions to wide networks. We extend and refine these results to obtain the dynamics for a wider class of λ-balanced networks with unequal input output dimension to more clearly demonstrate the impact of initialisation on *rich* and *lazy* learning regimes also developed in Tu et al. [280] for a set of orthogonal initalisations. Our work extends previous analyses [297] of these regimes to wide networks. Previous studies leveraged these solutions primarily to characterise convergence rates; however, our work goes beyond this by providing a comprehensive characterisation of the complete dynamics of the system [272].

2.2.3.2 Two-layer non-linear networks

Two-layer (or single-hidden-layer) networks have become a central model for studying the rich learning regime [307, 15, 75, 284, 196, 308, 242, 106, 304, 22]. A line of research, grounded in statistical mechanics, has developed analytically tractable dynamical models for a broad class of two-layer (or single-hidden-layer) artificial neural networks [304]. This approach typically involves taking asymptotic limits of large input dimension, number of samples and/or number of network weights in order to capture the typical case behaviour that shape learning behaviour. In these analyses, the network is characterised by how its width scales relative to the input dimension and the number of training samples, allowing for the exploration of distinct learning dynamics and representational properties across various scaling regimes [58].

Many recent advances in understanding the rich regime of deep learning have emerged from investigating how the variance of initialisation and the layer-wise learning rates should scale in the infinite-width limit, in order to ensure the consistent evolution of activations, gradients, and outputs. In this regime, analysing the dynamics becomes significantly more tractable: random variables tend to concentrate, and key quantities either vanish, remain constant, or diverge [180]. Specifically, the *mean-field limit* refers to the setting in which the input dimension remains finite while the number of hidden units tends to infinity [196, 260]. In contrast, the *extensive-width limit* describes a regime where the network width and input dimension scale proportionally with the number of training samples [58]. These framework enables the study of learning dynamics in overparameterised models without collapsing into the kernel regime, where no feature learning occurs [58]. In contrast, in the *Neural Tangent Kernel* (NTK) limit, the network behaves as a kernel regressor and does not engage in feature learning [130, 165, 16]. These ideas have been extended to deeper networks through the *tensor programme* framework, which has led to the derivation of the *maximal update parametrisation* (μP) [301, 300].

In Chapter 4 we focus on finite-width neural networks. The *finite-width* setting considers the case where the input dimension tends to infinity while the number of hidden units remains small. Theoretical analyses of these simpler models -such as generalised linear models (GLMs) without hidden layers or networks with a fixed (i.e., $\Theta(1)$) number of hidden units–have been extensively conducted [97, 287, 255, 263, 238], with comprehensive reviews available in [304]. In Chapter 4, we build on this framework to investigate the emergence of *specialisation* —a indication of the rich learning regime. Specialisation refers to the tendency of a network to rely on a subset of its hidden units, rather than redundantly distributing the representation across all neurons [26, 122]. Previous work [238, 106] has demonstrated that architectural choices and activation functions play a critical role in shaping the specialisation profile. We adopt this perspective to disentangle the respective contributions of initialisation and activation functions in determining the representation, with particular emphasis on how these factors influence specialisation.

2.2.3.3 Recurrent neural networks

Recurrent neural networks (RNNs) are important tools in both machine learning and neuroscience for learning tasks with temporal dependencies. In tandem with the success of dynamical systems theory in describing neural activity related to motor control, working memory, and decision-making [228, 289, 143], RNNs have become a popular choice for cognitive models of neural dynamics [25]. They not only replicate recurrent dynamics recorded in animals but are also capable of performing abstractions of the same cognitive tasks used in experiments [184, 83, 290, 187, 188]. More generally, RNNs are universal approximators of any open dynamical systems due to their time-evolving hidden layers, making them an appealing model for study [70, 254].

Accompanying the popularity of RNNs, there have been significant efforts dedicated to their theoretical understanding, both from deep learning theoreticians [54, 208, 314] and neuroscientists relating these findings to observations about the brain [189, 252, 253, 78, 88, 71, 176]. However, most theoretical studies of RNNs only analyse the end of training- analysing properties of the solutions they find, ignoring the learning process itself [241]. In Chapter 3, we examine simplified Linear Recurrent Neural Networks (LRNNs) to develop analytical methods for understanding their dynamics. These models have previously been used to investigate various phenomena, such as learning regimes [81, 176], low-rank modifications to network connectivity [252], and issues related to stability and extrapolation [54]. Differing from prior work on learning dynamics in linear networks [243, 244, 242, 39, 67, 240], we study recurrent architecture, allowing us to analyse how other architectures constrain optimisation in ways that differ from feedforward ones. Overall, despite the widespread use and known complex computational abilities of RNNs, it is still unknown how their representations emerge as a result of training on temporally-structured tasks.

2.2.4 Analytical frameworks

Below, we present two influential frameworks that have been introduced to the machine learning field to enable tractable theoretical analysis and deepen our understanding of learning dynamics. The first is the *teacher–student framework*, which offers a generative modelling setup for studying generalisation and internal representations in a controlled, synthetic setting. The second is the concept of *implicit bias*, which provides insights into the inductive principles that guide learning. Together, these frameworks offer complementary perspectives on how architectural choices, initialisation strategies, and training dynamics shape learning outcomes. In the subsections that follow, we explore each framework in detail and explain how they underpin the analyses presented in subsequent chapters.

2.2.4.1 Teacher–Student Framework

The teacher–student framework provides a generative modelling approach in which a student network learns from a teacher model, enabling the creation of controlled synthetic datasets. This setup offers a robust foundation for analysing learning behaviour under well-specified conditions [97]. Importantly, the framework can be adapted to different architectures and settings, making it a flexible and widely applicable tool across different model classes.

One of the principal advantages of this framework is its capacity to isolate and examine how internal representations relate to task structure within a systematically controlled setting. As a result, it has been widely adopted in both the machine learning and neuroscience communities to investigate generalisation and the development of internal representations [291, 82, 245, 311, 106, 19, 277, 220].

A substantial body of theoretical research has also explored online learning within this framework, particularly in the context of multilayer-perceptrons [34, 237, 106]. This line of work has facilitated the derivation of ordinary differential equations characterising the learning dynamics of the network, even in non-linear settings. While these equations often require numerical integration for their solution, they provide valuable insights into the behaviour of learning systems. More recently, this approach has been extended to the study of continual learning—where networks must integrate information across time without succumbing to catastrophic forgetting [17, 167, 168, 161].

In Chapter 4, we build upon this foundation by examining continual learning within the teacher–student framework [167, 168]. In particular, we focus on how different initialisation strategies influence the evolution of internal representations across sequential tasks.

2.2.4.2 Implicit bias

Modern deep learning models are often *overparameterised*—they contain far more parameters than there are training examples. According to classical statistical intuition, such models should easily overfit the training data, memorising it without learning patterns that generalise to unseen inputs. Yet, in practice, overparameterised networks often generalise surprisingly well. This counterintuitive behaviour is characterised by the phenomenon of *double descent* [28], where test error initially decreases with model complexity, then increases (as expected), but decreases again as the model becomes highly overparameterised. This suggests that, despite the existence of many interpolating solutions, neural networks are systematically biased toward those that generalise better. The source of this preference to converge to particular kinds of solutions is known as the network's *implicit bias*. The model architecture, initialisation, and optimisation procedure, in interaction with the data, determine the implicit bias. Understanding this bias is crucial not only for explaining why neural networks generalise well despite their capacity to overfit but also for informing principled model design and training strategies.

Recent progress in understanding implicit bias has centred on theoretically characterising how gradient-based optimisation selects specific solutions from the multitude that perfectly fit the training data. A prevalent approach involves identifying a function Q such that the network converges to a first-order Karush-Kuhn-Tucker (KKT) point that minimises Q among all interpolating solutions. Foundational work by Soudry et al. [265] exemplified this framework by demonstrating that linear classifiers trained with gradient descent converge to the maximum-margin solution. This concept has since been extended to various architectures, including deep linear networks [137, 112, 199], homogeneous networks [181, 202, 50], and recurrent neural networks [81].

A related line of research examines the learning dynamics of networks trained with mean squared error, where the network function dynamics can be expressed as a *mirror flow* governed by a specific potential function $\Phi(\beta)$. Mirror flow is a type of continuous-time optimisation that operates not in the original parameter space (known as the primal space), but in a transformed, dual space—often referred to as the *mirror space*. Instead of directly updating the model parameters using gradients in the primal space, this method updates a transformed version of the parameters, typically based on the gradient of the potential function. This approach is particularly effective in settings with non-Euclidean geometries or constraints, as the choice of the potential function $\Phi(\beta)$ defines the geometry underlying the optimisation process. Within this framework, the function Q, introduced earlier as a measure of implicit bias, emerges directly from these dynamics and can be characterised as minimising a *Bregman divergence* associated with $\Phi(\beta)$. The Bregman divergence captures how far a solution is from optimality with respect to the geometry defined by $\Phi(\beta)$. Consequently, the mirror flow dynamics implicitly guide the optimisation toward solutions that minimise Q by following paths of steepest descent in this geometry. This perspective provides a natural geometric interpretation of implicit bias and helps explain why mirror descent-like processes tend to favour certain interpolating solutions over others [111]. More details on the mirror flow analysis can be found in Appendix A.3.5.1.

In simple models like diagonal linear networks, this method shows how the bias shifts between favouring sparse (ℓ^1-like) and smooth (ℓ^2-like) solutions depending on the training regime [294]. However, finding the potential $\Phi(\beta)$ is problem-specific and requires solving a second-order differential equation, which may not be solvable even in simple settings [113, 171]. Azulay et al.[21] extended this analysis to a time-warped mirror flow, enabling the study of a broader class of architectures (finding a solution for the second-order differential equation). In Chapter 3, we build on this line of research by deriving exact expressions for the inductive bias in our minimal linear network model. Our results extend the work by Azulay et al. [21] to wide linear networks, revealing how structure and initialisation influence the solutions found during training.

Chapter 3

From Lazy to Rich: Exact learning dynamics in deep linear networks with prior knowledge

Finally, we make some remarks on why linear systems are so important.
The answer is simple: because we can solve them!
– Richard Feynman

This chapter discusses the work presented in publications [67, 69, 158] as well as ongoing research on Linear RNNs and a review on Linear models [203, 219].

3.1 Introduction

The success of neural models relies on their ability to extract relevant features from data to build internal representations, a complex process that in machine learning is defined by two regimes: *lazy* and *rich* [243, 215, 51, 22]. Despite significant advances, these learning regimes and their characterisation are not yet fully understood and would benefit from clearer theoretical predictions, particularly regarding the influence of prior knowledge on the learning regime. Here, prior knowledge can be interpreted either as the initialisation or as the representation learned from a previous task, as is common in continual learning. In this work, we primarily focus on the impact of initialisation, which can be thought of as the original weight structure—shaped by years of evolution—potentially embedding useful priors about the types of tasks the network is expected to encounter and solve. In this work, we address this gap by deriving exact solutions for the learning dynamics in deep linear networks as a function of network initialisation, providing one of the few analytical models of the *rich* and *lazy* regimes in wide and deep neural networks [297, 158, 280].

To illustrate the dramatic effect of initialisation and the kind of phenomenon we build a theory for, we consider a two-layer linear network parameterised by an encoding layer $\mathbf{W}_1$ and a decoding layer $\mathbf{W}_2$ (Fig. 3.1A). This network can be initialised with different relative scalings, such that $\mathbf{W}_1\mathbf{W}_1^T \succ \mathbf{W}_2^T\mathbf{W}_2$, $\mathbf{W}_1\mathbf{W}_1^T \prec \mathbf{W}_2^T\mathbf{W}_2$, or $\mathbf{W}_1\mathbf{W}_1^T = \mathbf{W}_2^T\mathbf{W}_2$, while maintaining the same absolute scale - where ee define the absolute scale of the weights as the norm of $\mathbf{W}_2\mathbf{W}_1$ [1]. As shown in Fig. 3.1B, the choice of relative scaling can result in drastically different learning trajectories and representations and the theory we develop over the course of this paper describes these effects. We will exactly solve the dynamics for a special case in Section 3.3. Through these solutions, we aim to gain insights into the *rich* and *lazy* regimes, as well as the transition between them during training, by examining the impact of relative scaling. As shown in Fig. 3.1C and further proved in Appendix A.1.1, initialisation methods used in practice, such as LeCun initialisation in wide networks, approximate the relative scaling initialisation explored in this paper, making it relevant to machine learning community as further demonstrated by Kunin et al. [158]. We consider applications relevant to machine learning and neuroscience, including continual learning [148, 306, 212], reversal learning [84], transfer learning [273, 276, 161, 101], fine-tuning [124], revising structured knowledge [247] and prioritised replay for revising structured knowledge [191].

Our contributions.

- We derive explicit solutions for the gradient flow, internal representational similarity, and finite-width NTK in unequal-input-output two-layer deep linear networks under a broad range of λ-balanced initialisation conditions.

[1]the symbols $\succ$ and $\prec$ refer to positive definite matrix inequalities.

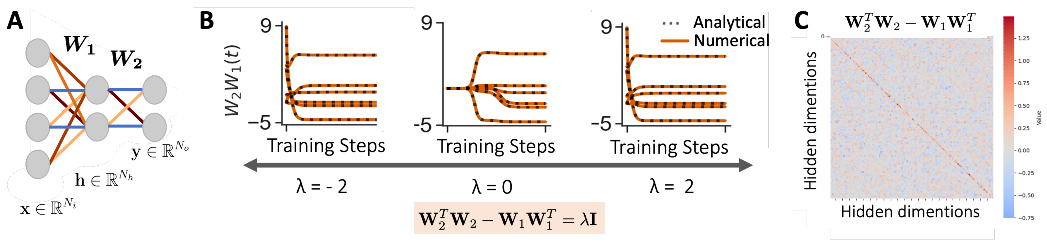


Figure 3.1: A minimal model of the *rich* and *lazy* regimes. **A.** We examine a deep and wide linear network trained using gradient descent starting from an initialisation characterised by a relative scale parameter λ — which characterises the difference in pairwise products between the first and second layers ($\mathbf{W}_2^T\mathbf{W}_2 - \mathbf{W}_1\mathbf{W}_1^T$). **B.** Network output for an example task over training time, starting from a range of relative scale values. The dynamics are influenced by the initialisation. Solid lines represent simulations, while dotted lines indicate the analytical solutions derived in this work. **C.** A network with LeCun weight initialisation [164] in the infinite width limit becomes λ-balanced, as $\mathbf{W}_2^T\mathbf{W}_2 - \mathbf{W}_1\mathbf{W}_1^T$ approaches the scaled identity matrix. For simulation details, see Appendix A.5.3.

- We model the full range of learning dynamics from *lazy* to *rich*, showing that this transition is influenced by a complex interaction of architecture, *relative scale*, *absolute scale* and *weight-target ratio* extending beyond just initialisation *absolute scale*. We further analyse how weights dynamically align to task-relevant structure over the course of learning, going beyond prior work that has assumed initial alignment.

- We present applications of these solutions relevant to both the neuroscience and machine learning communities, providing exact solutions for continual learning dynamics, reversal learning dynamics, fine-tuning, and learning structured knowledge and application to transfer learning.

3.2 Preliminaries

Consider a supervised learning task where input vectors $\mathbf{x}_n \in \mathbb{R}^{N_i}$, from a set of P training pairs $\{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^{P}$, need to be mapped to their corresponding target output vectors $\mathbf{y}_n \in \mathbb{R}^{N_o}$. We learn this task with a two-layer linear network model

$$\hat{\mathbf{y}}_n = \mathbf{W}_2 \mathbf{W}_1 \mathbf{x}_n, \tag{3.1}$$

with weight matrices $\mathbf{W}_1 \in \mathbb{R}^{N_h \times N_i}$ and $\mathbf{W}_2 \in \mathbb{R}^{N_o \times N_h}$, where N_h is the number of hidden units. The network's weights are optimised using full batch gradient descent with learning rate η (or respectively time constant $\tau = \frac{1}{\eta}$) on the mean squared error loss

$$\mathcal{L}(\hat{\mathbf{y}}, \mathbf{y}) = \frac{1}{2} \left\langle ||\hat{\mathbf{y}} - \mathbf{y}||^2 \right\rangle, \tag{3.2}$$

where $\langle \cdot \rangle$ denotes the average over the dataset. Our objective is to describe the entire dynamics of the network's output and internal representations based on the input covariance and input-output cross-covariance matrices of the dataset, defined as

$$\tilde{\mathbf{\Sigma}}^{xx} = \frac{1}{P} \sum_{n=1}^{P} \mathbf{x}_n \mathbf{x}_n^T \in \mathbb{R}^{N_i \times N_i} \quad \text{and} \quad \tilde{\mathbf{\Sigma}}^{yx} = \frac{1}{P} \sum_{n=1}^{P} \mathbf{y}_n \mathbf{x}_n^T \in \mathbb{R}^{N_o \times N_i}, \tag{3.3}$$

and the initialisation $\mathbf{W}_2(0), \mathbf{W}_1(0)$. We employ an approach first introduced in the foundational work of Fukumizu [93] (discribed in Appendix A.1.2), which, instead of studying the parameters directly, considers the dynamics of a matrix of the important statistics. In particular, defining $\mathbf{Q} = \begin{bmatrix} \mathbf{W}_1 & \mathbf{W}_2^T \end{bmatrix}^T \in \mathbb{R}^{(N_i+N_o) \times N_h}$, we consider the $(N_i + N_o) \times (N_i + N_o)$ matrix

$$\mathbf{Q}\mathbf{Q}^T(t) = \begin{bmatrix} \mathbf{W}_1^T \mathbf{W}_1(t) & \mathbf{W}_1^T \mathbf{W}_2^T(t) \\ \mathbf{W}_2 \mathbf{W}_1(t) & \mathbf{W}_2 \mathbf{W}_2^T(t) \end{bmatrix}, \tag{3.4}$$

which is divided into four quadrants with interpretable meanings, and where $t \in \mathbb{R}$ represents training time. The approach monitors several key statistics collected in the matrix. The off-diagonal blocks contain the network function $\hat{\mathbf{Y}}(t) = \mathbf{W}_2 \mathbf{W}_1(t) \mathbf{X}$. The on-diagonal blocks capture the correlation structure of the weight matrices, allowing for the calculation of the temporal evolution of the network's internal representations. This includes the representational similarity matrices (RSM) of the neural representations within the hidden layer,

$$\mathrm{RSM}_I = \mathbf{X}^T \mathbf{W}_1^T \mathbf{W}_1(t)\mathbf{X}, \quad \mathrm{RSM}_O = \mathbf{Y}^T (\mathbf{W}_2 \mathbf{W}_2^T(t))^+ \mathbf{Y}, \tag{3.5}$$

where $+$ denotes the pseudoinverse; and the network's finite-width NTK [130, 165, 16]

$$\mathrm{NTK} = \mathbf{I}_{N_o} \otimes \mathbf{X}^T \mathbf{W}_1^T \mathbf{W}_1(t)\mathbf{X} + \mathbf{W}_2 \mathbf{W}_2^T(t) \otimes \mathbf{X}^T \mathbf{X}, \tag{3.6}$$

where $\mathbf{I}_{N_o}$ is the $N_o \times N_o$ identity matrix and $\otimes$ is the Kronecker product. Hence, the dynamics of $\mathbf{Q}\mathbf{Q}^T$ describes the important aspects of network behaviour. For a derivation of these quantities, see Appendix A.1.3.

3.3 Exact learning dynamics

In this section, we derive an exact solution for $\mathbf{QQ}^T$, providing a clean understanding of the learning dynamics, convergence behaviour, and generalisation properties of two-layer linear networks with prior knowledge.

3.3.1 Assumptions

To derive these solutions, we make the following assumptions:

- **A1** (*Whitened input*). The input data is whitened, i.e. $\tilde{\mathbf{\Sigma}}^{xx} = \mathbf{I}$.

- **A2** (*λ-Balanced*). The network's weight matrices are λ-balanced at the beginning of training, i.e. $\mathbf{W}_2^T\mathbf{W}_2(0) - \mathbf{W}_1\mathbf{W}_1(0)^T = \lambda\mathbf{I}$. If this condition holds at initialisation, it will persist throughout training [243, 12]. For completeness, we prove this in Appendix A.2.1.

- **A3** (*Dimensions*). The hidden dimension of the network is defined as $N_h = \min(N_i, N_o)$, ensuring the network is neither bottlenecked ($N_h < \min(N_i, N_o)$) nor overparameterised ($N_h > \min(N_i, N_o)$).

These assumptions are strictly weaker than prior works [93, 158, 297]. The main distinction between our work and prior works is that [93] assumed zero-balanced weights ($\mathbf{W}_1(0)\mathbf{W}_1(0)^T = \mathbf{W}_2(0)^T\mathbf{W}_2(0)$), while we relax this assumption to λ-balanced. The zero-balanced condition restricts the networks to a *rich* setting. We develop solutions to explore the continuum between the *rich* and the *lazy* regime. Furthermore, we develop solutions to be applicable to equal and unequal input and output dimensions, overcoming assumptions used in Fukumizu et al. [93]. While some works, such as Tarmoun et al. [272], have considered removing this constraint, their solutions remain in an unstable and mixed form. Other studies, such as Xu and Ziyin [297] and Kunin et al. [158], have similarly relaxed the balanced assumption but were limited to single-output neuron settings. See Appendix A.2.2 for a further discussion on each of these works' assumptions and their relationship to ours.

3.3.2 Lemmas and definitions

To derive exact solutions, we start by presenting the main lemmas, which we prove in the Appendix.

Lemma 3.3.1. *Under Assumptions 1 and 2, the gradient flow dynamics of* $\mathbf{QQ}^T(t)$*, with initialisation* $\mathbf{QQ}^T(0) = \mathbf{Q}(0)\mathbf{Q}(0)^T$ *can be written as a differential matrix Riccati equation*

$$\tau\frac{d}{dt}(\mathbf{QQ}^T) = \mathbf{FQQ}^T + \mathbf{QQ}^T\mathbf{F} - (\mathbf{QQ}^T)^2, \quad \textit{where } \mathbf{F} = \begin{pmatrix} -\frac{\lambda}{2}\mathbf{I}_{N_i} & (\tilde{\mathbf{\Sigma}}^{yx})^T \\ \tilde{\mathbf{\Sigma}}^{yx} & \frac{\lambda}{2}\mathbf{I}_{N_o} \end{pmatrix}. \tag{3.7}$$

As derived in [93], whenever $\mathbf{F}$ is symmetric and diagonalisable such that $\mathbf{F} = \mathbf{P\Lambda P}^T$, where $\mathbf{P}$ is an orthonormal matrix and $\mathbf{\Lambda}$ is a diagonal matrix, then the unique solution to this matrix Riccati equation is given by

$$\mathbf{QQ}^T(t) = e^{\mathbf{F}\frac{t}{\tau}}\mathbf{Q}(0)\left[\mathbf{I} + \mathbf{Q}(0)^T\mathbf{P}\left(\frac{e^{2\mathbf{\Lambda}\frac{t}{\tau}} - \mathbf{I}}{2\mathbf{\Lambda}}\right)\mathbf{P}^T\mathbf{Q}(0)\right]^{-1}\mathbf{Q}(0)^T e^{\mathbf{F}\frac{t}{\tau}}. \tag{3.8}$$

In Appendix A.2.3, we prove that this equation is the unique solution to the initial value problem derived in Lemma 3.3.1 for any value of Λ. However, the solution in this form is not very useable or interpretable due to the matrix inverse mixing the blocks of $\mathbf{QQ}^T$. Additionally, we need to diagonalise $\mathbf{F}$. To do so we consider the compact singular value decomposition $\text{SVD}(\tilde{\mathbf{\Sigma}}^{yx}) = \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T$. Here, $\tilde{\mathbf{U}} \in \mathbb{R}^{N_o \times N_h}$ denotes the left singular vectors, $\tilde{\mathbf{S}} \in \mathbb{R}^{N_h \times N_h}$ the square matrix with ordered, non-zero singular values on its diagonal, and $\tilde{\mathbf{V}} \in \mathbb{R}^{N_i \times N_h}$ the corresponding right singular vectors. For unequal input-output dimensions ($N_i \neq N_o$), the right and left singular vectors are not square. Accordingly, for the case $N_i > N_h = N_o$, we define $\tilde{\mathbf{U}}_\perp \in \mathbb{R}^{N_o \times |N_o - N_i|}$ as a matrix containing orthogonal column vectors that complete the basis for $\tilde{\mathbf{U}}$, i.e., make $\left[\tilde{\mathbf{U}}\ \tilde{\mathbf{U}}_\perp\right]$ orthonormal, and $\tilde{\mathbf{V}}_\perp \in \mathbb{R}^{N_i \times |N_o - N_i|}$ as a matrix of zeros. Conversely, when $N_i = N_h < N_o$, then $\tilde{\mathbf{V}}_\perp$ is a matrix containing orthogonal column vectors that complete the basis for $\tilde{\mathbf{V}}$ and $\tilde{\mathbf{U}}_\perp$ is a matrix of zeros. Using this SVD structure, we can now describe the eigendecomposition of $\mathbf{F}$ (Proof in Appendix A.2.4).

Lemma 3.3.2. *Under Assumption 3, the eigendecomposition of* $\mathbf{F} = \mathbf{P}\boldsymbol{\Lambda}\mathbf{P}^T$ *is*

$$\mathbf{P} = \frac{1}{\sqrt{2}} \begin{pmatrix} \tilde{\mathbf{V}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \tilde{\mathbf{V}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \sqrt{2}\tilde{\mathbf{V}}_\perp \\ \tilde{\mathbf{U}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & -\tilde{\mathbf{U}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \sqrt{2}\tilde{\mathbf{U}}_\perp \end{pmatrix}, \quad \boldsymbol{\Lambda} = \begin{pmatrix} \tilde{\mathbf{S}}_\lambda & 0 & 0 \\ 0 & -\tilde{\mathbf{S}}_\lambda & 0 \\ 0 & 0 & \boldsymbol{\lambda}_\perp \end{pmatrix}, \tag{3.9}$$

where the matrices $\tilde{\mathbf{S}}_\lambda$*,* $\lambda_\perp$*,* $\tilde{\mathbf{H}}$*, and* $\tilde{\mathbf{G}}$ *are diagonal matrices defined as:*

$$\tilde{\mathbf{S}}_\lambda = \sqrt{\tilde{\mathbf{S}}^2 + \frac{\lambda^2}{4}\mathbf{I}}, \; \lambda_\perp = \operatorname{sgn}(N_o - N_i)\frac{\lambda}{2}\mathbf{I}_{|N_o - N_i|}, \; \tilde{\mathbf{H}} = \operatorname{sgn}(\lambda)\sqrt{\frac{\tilde{\mathbf{S}}_\lambda - \tilde{\mathbf{S}}}{\tilde{\mathbf{S}}_\lambda + \tilde{\mathbf{S}}}}, \; \tilde{\mathbf{G}} = \frac{1}{\sqrt{\mathbf{I} + \tilde{\mathbf{H}}^2}}. \tag{3.10}$$

3.3.3 Main theorem

Thanks to the eigendecomposition of $\mathbf{F}$, we can separate the solution provided in Eq. 3.8 into four quadrants. The following variables of initialisation allow us to define the product $\mathbf{P}^T\boldsymbol{Q}(0)$ more succinctly,

$$\mathbf{B} = \mathbf{W}_2(0)^T\tilde{\mathbf{U}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}}) + \mathbf{W}_1(0)\tilde{\mathbf{V}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}}) \in \mathbb{R}^{N_h \times N_h}, \tag{3.11}$$

$$\mathbf{C} = \mathbf{W}_2(0)^T\tilde{\mathbf{U}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}}) - \mathbf{W}_1(0)\tilde{\mathbf{V}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}}) \in \mathbb{R}^{N_h \times N_h}, \tag{3.12}$$

$$\mathbf{D} = \mathbf{W}_2(0)^T\tilde{\mathbf{U}}_\perp + \mathbf{W}_1(0)\tilde{\mathbf{V}}_\perp \in \mathbb{R}^{N_h \times |N_o - N_i|}. \tag{3.13}$$

Using these variables of the initialisation, this brings us to our main theorem:

Theorem 3.3.3. *Under the assumptions of whitened inputs (1),* λ*-balanced weights (2), and no bottleneck (3), the temporal dynamics of* $\mathbf{Q}\mathbf{Q}^T$ *are*

$$\mathbf{Q}\mathbf{Q}^T(t) = \begin{pmatrix} \mathbf{Z_1}(t)\mathbf{A}^{-1}(t)\mathbf{Zx_1}^T(t) & \mathbf{Z_1}(t)\mathbf{A}^{-1}(t)\mathbf{Z_2}^T(t) \\ \mathbf{Z_2}(t)\mathbf{A}^{-1}(t)\mathbf{Z_1}^T(t) & \mathbf{Z_2}(t)\mathbf{A}^{-1}(t)\mathbf{Z_2}^T(t) \end{pmatrix},$$

with the time-dependent variables $\mathbf{Z_1}(t) \in \mathbb{R}^{N_i \times N_h}$*,* $\mathbf{Z_2}(t) \in \mathbb{R}^{N_o \times N_h}$*, and* $\mathbf{A}(t) \in \mathbb{R}^{N_h \times N_h}$*:*

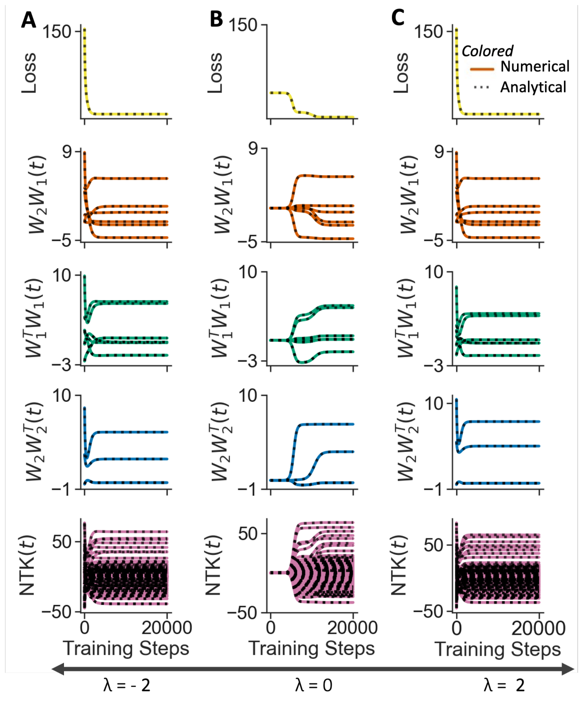


Figure 3.2: **A.** The temporal dynamics of the numerical simulation (colored lines) of the loss, network function, correlation of input and output weights, and the NTK (rows 1-5, respectively) are exactly matched by the analytical solution (black dotted lines) for $\lambda = -2$. **B.** $\lambda = 0$ Large initial weight values. **C.** $\lambda = 2$ initial weight values initialised as described in A.5. For simulation details, see Appendix A.5.4.

$$\mathbf{Z_1}(t) = \frac{1}{2}\tilde{\mathbf{V}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^T - \frac{1}{2}\tilde{\mathbf{V}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T + \tilde{\mathbf{V}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\mathbf{D}^T, \tag{3.14}$$

$$\mathbf{Z_2}(t) = \frac{1}{2}\tilde{\mathbf{U}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^T + \frac{1}{2}\tilde{\mathbf{U}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T + \tilde{\mathbf{U}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\mathbf{D}^T, \tag{3.15}$$

$$\mathbf{A}(t) = \mathbf{I} + \mathbf{B}\left(\frac{e^{2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} - \mathbf{I}}{4\tilde{\mathbf{S}}_\lambda}\right)\mathbf{B}^T - \mathbf{C}\left(\frac{e^{-2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} - \mathbf{I}}{4\tilde{\mathbf{S}}_\lambda}\right)\mathbf{C}^T + \mathbf{D}\left(\frac{e^{\lambda_\perp \frac{t}{\tau}} - \mathbf{I}}{\lambda_\perp}\right)\mathbf{D}^T. \tag{3.16}$$

The proof of Theorem 3.3.3 is in Appendix A.2.5. With this solution, we can calculate the exact temporal dynamics of the loss, network function, RSMs and NTK (Fig. 3.2A, C) over a range of λ-balanced initialisations.

3.3.4 Implementation and simulation

One issue with the expression we derived in Theorem 3.3.3 is that it can be numerically unstable when simulating it for a long time $t \gg 0$ as it involves taking the inverse of terms that involve exponentials that are diverging with t. If we make the additional assumption that $\mathbf{B}$ is invertible, then we can rearrange this expression to only use exponentials with negative coefficients, which we derive in Appendix A.2.6. In the next section, we will discuss the significance of $\mathbf{B}$ being invertible at initialisation on the convergence of the dynamics. Simulation details are in Appendix A.5.

3.3.5 Lambda zero

Since the inverse $\mathbf{F}^{-1} = \mathbf{P}\mathbf{\Gamma}^{-1}\mathbf{P}^T$ is not well-defined when $\mathbf{\Gamma}$ has zero singular values, we analyse singular values of zero by examining the limiting behaviour using L'Hôpital's rule. The method is described in Appendix A.2.7. In this setting, we impose a full-rank initialisation condition, defined as $\text{rank}(\mathbf{W}_2(0)\mathbf{W}_1(0)) = N_i = N_o$. However, this assumption is not necessary in our framework when lambda is non-zero.

3.4 Rich and Lazy learning

The conserved quantity λ - set at initialisation - arises from an underlying symmetry in the network, reminiscent of Noether's theorem, where symmetries correspond to conserved quantities. In this case, the weights $\mathbf{W_1}$ and $\mathbf{W_2}$ can be transformed via an invertible matrix $\mathbf{G}$ and its inverse $\mathbf{G}^{-1}$, respectively, without altering the function computed by the network—i.e., $\mathbf{W_2}\mathbf{W_1}$ remains unchanged. This transformation, $\mathbf{W_1} \to \mathbf{G}\mathbf{W_1}$ and $\mathbf{W_2} \to \mathbf{G}^{-1}\mathbf{W_2}$, constitutes a General Linear (GL) transformation of the network's internal representation. Although such GL transformations leave the network's output unchanged, standard gradient descent is not equivariant under them. This lack of invariance is what drives the emergence of different learning regimes. As a result, the learning dynamics become highly sensitive to the relative scaling of layers, encapsulated by λ. In contrast to GL-equivariant optimisation methods—such as Newton's method, which accounts for curvature and is therefore more robust to reparameterisations—gradient descent breaks this symmetry. Ultimately, this asymmetry originates in the backpropagation rule itself, where gradients at each layer are scaled by the weights of subsequent layers—providing an intuitive explanation for how the learning dynamics evolve with λ. In summary, the *relative scale/ balanced condition* introduces an asymmetry in the update rule that leads to distinct learning regimes.

In this section, we study how the learning dynamics vary with λ, the *relative scale/ balanced condition*, as it spans from negative to positive infinity. We leverage the exact solutions to gain deeper insight into the transition between the rich and lazy learning regimes, analysing how this transition depends on the level of prior knowledge, specifically the initialisation. We investigate five key indicators of learning regimes: the dynamics of singular values and vectors, the behaviour, structure, and robustness of learned representations, and the evolution of the NTK. We also examine the influence of the initialisation scale and architectural choices on learning dynamics. These analyses incorporate new analytical methods that do not build on the previously presented solution but provide a complementary framework that goes beyond the assumptions required for that solution, offering a fresh perspective on the problem.

3.4.1 Dynamics of the singular values

Here, we study a λ-*balanced* linear network initialised with *task-aligned* weights. Task-aligned weights are those initialised to share the same eigenvectors as the target task, allowing us to focus solely on the dynamics of the corresponding singular values. Previous research [246] has demonstrated that initial weights that are aligned with the task remain aligned throughout training, restricting the learning dynamics to the singular values of the network. The λ-*balanced* setting offers a valuable opportunity to build intuition about the impact of imbalance on the dynamics of learning regimes, extending beyond previous solutions [272, 285].

Theorem 3.4.1. *Under the assumptions of Theorem 3.3.3 and with a task-aligned initialisation, as defined in [243], the network function is given by the expression* $\mathbf{W}_2\mathbf{W}_1(t) = \tilde{\mathbf{U}}\mathbf{S}(t)\tilde{\mathbf{V}}^T$ *where* $\mathbf{S}(t) \in \mathbb{R}^{N_h \times N_h}$ *is a diagonal matrix of singular values with elements* $s_\alpha(t)$ *that evolve according to the equation,*

$$s_\alpha(t) = s_\alpha(0) + \gamma_\alpha(t;\lambda)\left(\tilde{s}_\alpha - s_\alpha(0)\right), \tag{3.17}$$

where $\tilde{s}_\alpha$ *is the* α *singular value of* $\tilde{\mathbf{S}}$ *and* $\gamma_\alpha(t;\lambda)$ *is a* λ*-dependent monotonic transition function for each singular value that increases from* $\gamma_\alpha(0;\lambda) = 0$ *to* $\lim_{t\to\infty}\gamma_\alpha(t;\lambda) = 1$ *defined explicitly in Appendix A.3.1. We find that under different limits of* λ*, the transition function converges pointwise to the sigmoidal (*$\lambda \to 0$*) and exponential (*$\lambda \to \pm\infty$*) transition functions,*

$$\lim_{\lambda\to 0}\gamma_\alpha(t;\lambda) \to \frac{e^{2\tilde{s}_\alpha\frac{t}{\tau}} - 1}{e^{2\tilde{s}_\alpha\frac{t}{\tau}} - 1 + \frac{\tilde{s}_\alpha}{s_\alpha(0)}}, \qquad \lim_{\lambda\to\pm\infty}\gamma_\alpha(t;\lambda) \to 1 - e^{-|\lambda|\frac{t}{\tau}}. \tag{3.18}$$

The proof for Theorem 3.4.1 can be found in Appendix A.3.1. As shown in Fig. 3.3B, as λ approaches zero, the dynamics resemble sigmoidal learning curves that traverse between saddle points, characteristic of the *rich* regime. In this

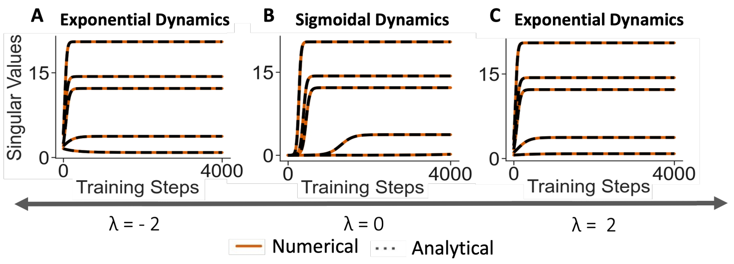


Figure 3.3: Simulated and analytical dynamics of the singular values of the network function with *relative scale* of **A.** $\lambda = -2$, **B.** $\lambda = 0$, or **C.** $\lambda = 2$, initialised as described in Appendix A.5. For simulation details, see Appendix A.5.5.

regime, the network learns the most salient features first, which can be beneficial for generalisation [161]. Conversely, as shown in Fig. 3.3A and C, as the magnitude of λ increases, the dynamics become exponential, characteristic of the *lazy* regime. In this regime, all features are treated equally and the network's dynamics resemble that of a shallow network. Notably, similar effects have been observed previously in the context of large *absolute scales* [246] independently of the *relative scale*. Overall, our results highlight the critical influence the *relative scale* λ has in shaping the learning dynamics, from sigmoidal to exponential, steering the network between the *rich* and *lazy* regimes.

3.4.2 The dynamics of the representations

We now consider how the representations of the individual parameters $\mathbf{W}_1$ and $\mathbf{W}_2$ change through training. Under λ-balanced initialisation, a simple structure persists throughout training, allowing us to recover the dynamics of parameters - up to a time-dependent orthogonal transformation - from that of $\mathbf{QQ}^T(t)$.

Theorem 3.4.2. *Under Assumption 2, if the network function* $\mathbf{W}_2\mathbf{W}_1(t) = \mathbf{U}(t)\mathbf{S}(t)\mathbf{V}^T(t)$ *is full rank, then we can recover the parameters* $\mathbf{W}_2(t) = \mathbf{U}(t)\mathbf{S}_2(t)\mathbf{R}^T(t)$ *and* $\mathbf{W}_1(t) = \mathbf{R}(t)\mathbf{S}_1(t)\mathbf{V}^T(t)$ *up to time-dependent orthogonal transformation* $\mathbf{R}(t) \in \mathbb{R}^{N_h \times N_h}$*, where* $\mathbf{S}_\lambda(t) = \sqrt{\mathbf{S}^2(t) + \frac{\lambda^2}{4}\mathbf{I}}$ *and*

$$\mathbf{S}_1(t) = \left(\left(\mathbf{S}_\lambda(t) - \frac{\lambda\mathbf{I}}{2}\right)^{\frac{1}{2}}, 0_{\max(0,N_i-N_o)}\right), \mathbf{S}_2(t) = \left(\left(\mathbf{S}_\lambda(t) + \frac{\lambda\mathbf{I}}{2}\right)^{\frac{1}{2}}; 0_{\max(0,N_o-N_i)}\right). \tag{3.19}$$

Proof of Theorem 3.4.2 is in Appendix A.3.2.

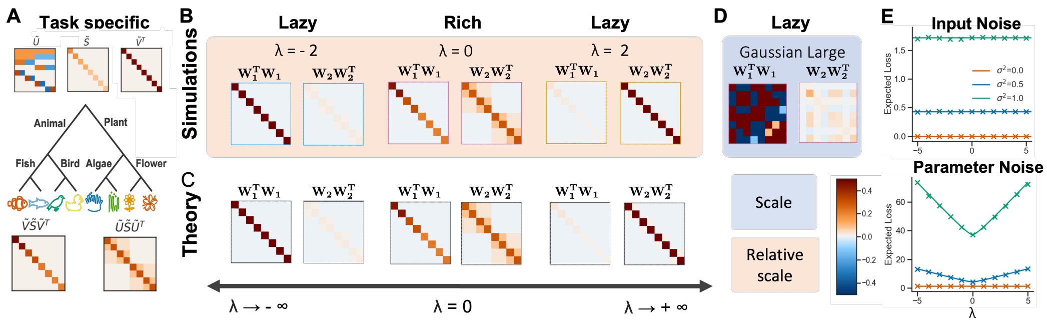


Figure 3.4: A. A semantic learning task with the SVD of the input-output correlation matrix of the task. (top) U and V represent the singular vectors, and S contains the singular values. (bottom) The respective RSMs are $USU^\top$ for the input and $VSV^\top$ for the output task. **B.** Simulation results and **C.** Theoretical input and output representation matrices after training, showing convergence when initialised with values of λ equal to -2, 0, and 2, according to the initialisation scheme described in Appendix A.5. **D.** Final RSMs matrices after training converged when initialised from random large weights. **E.** After convergence, the network's sensitivity to input noise (top panel) is invariant to λ, but the sensitivity to parameter noise increases as λ becomes smaller (or larger) than zero. For simulation details, see Appendix A.5.6.

The effective singular values $\mathbf{S}_\lambda$ of the corresponding weights are either up-weighted or down-weighted depending on the magnitude and sign of λ, splitting the representation into two parts. This division is reflected in the network's internal representations. With our solution, $\mathbf{QQ}^T(t)$, which captures the temporal dynamics of the similarity between hidden layer activations, we can analyse the network's internal representations in relation to the task. This allows us to determine whether the network adopts a *rich* or *lazy* representation, depending on the value of λ. Assuming convergence to the global minimum, which is guaranteed when the matrix $\mathbf{B}$ is non-singular, the internal representation satisfies $\mathbf{W}_1^T\mathbf{W}_1 = \tilde{\mathbf{V}}\tilde{\mathbf{S}}_1^2\tilde{\mathbf{V}}^T$ and $\mathbf{W}_2\mathbf{W}_2^T = \tilde{\mathbf{U}}\tilde{\mathbf{S}}_2^2\tilde{\mathbf{U}}^T$ with $\mathbf{W}_2\mathbf{W}_1 = \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T$. Theorem A.3.3 in the Appendix provides a detailed proof of this limiting behaviour.

To illustrate this, we consider a hierarchical semantic learning task[1], introduced in Saxe et al. [243], where living organisms are organised according to their features

[1]In this setting, the network has equal input and output dimensions

(Fig.3.4 A). The representational similarity of the task's inputs ($\tilde{\mathbf{V}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T$) reflects this hierarchical structure (Fig. 3.4A). Similarly, the representational similarity of the task's target values ($\tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{U}}^T$) highlights the primary groupings of items.

When training a two-layer network with *relative scale* λ equal to zero and task-agnostic initialisation [197], the input and output representational similarity matrices (Fig. 3.4B) match the task's structure upon convergence. As derived in Theorem A.3.4 the network is guaranteed to find a *rich* solution regardless of the *absolute scale* when lambda is equal to zero, meaning $\mathbf{W}_1^T\mathbf{W}_1 = \tilde{\mathbf{V}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T$ and $\mathbf{W}_2\mathbf{W}_2^T = \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{U}}^T$, as shown in Fig. 3.4C.

We also show that as λ approaches either positive or negative infinity, the network symmetrically transitions into the *lazy* regime.

Theorem 3.4.3. *Under the assumptions of Theorem A.2.6, training on data $\mathbf{\Sigma}^{yx} = \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T$, as $\lambda \to \infty$, the representation tends to*

$$\mathbf{W}_2\mathbf{W}_2^T = \tilde{\mathbf{U}} \begin{bmatrix} \lambda\mathbf{I} & 0_{\max(0,N_o-N_i)} \\ 0_{\max(0,N_o-N_i)} & 0 \end{bmatrix} \tilde{\mathbf{U}}^T, \tag{3.20}$$

$$\mathbf{W}_1^T\mathbf{W}_1 = \frac{1}{\lambda}\tilde{\mathbf{V}} \begin{bmatrix} \tilde{\mathbf{S}}^2 & 0_{\max(0,N_i-N_o)} \\ 0_{\max(0,N_i-N_o)} & 0 \end{bmatrix} \tilde{\mathbf{V}}^T. \tag{3.21}$$

As $\lambda \to -\infty$,

$$\mathbf{W}_2\mathbf{W}_2^T = -\frac{1}{\lambda}\tilde{\mathbf{U}} \begin{bmatrix} \tilde{\mathbf{S}}^2 & 0_{\max(0,N_o-N_i)} \\ 0_{\max(0,N_o-N_i)} & 0 \end{bmatrix} \tilde{\mathbf{U}}^T, \tag{3.22}$$

$$\mathbf{W}_1^T\mathbf{W}_1 = \tilde{\mathbf{V}} \begin{bmatrix} -\lambda\mathbf{I} & 0_{\max(0,N_i-N_o)} \\ 0_{\max(0,N_i-N_o)} & 0 \end{bmatrix} \tilde{\mathbf{V}}^T. \tag{3.23}$$

The proof can be found in Appendix A.3.2.3

As demonstrated in Theorem 3.4.3 and illustrated in Fig. 3.4B, the representations converge to an identity matrix for both large positive and large negative values of λ— emerging in the output representations for large positive λ and input representations for large negative λ. This convergence indicates that the network adopts task-agnostic representations. Meanwhile, the other respective RSMs become negligible, with scales proportional to $1/\lambda$, while still solving for the task. Therefore, as shown in Theorem A.3.6, the NTK becomes static and equivalent to the identity matrix in the limit as λ approaches infinity. However, the downscaled representations of the network retain task-specific structure. As demonstrated in Theorem 3.4.3, the downscaled weights captures the singular values of the task as well as the corresponding singular vectors.

Intuitively, in this setup, the larger weights function as an identity-like projection, while the smaller weights adapt and align to the task. However, because of their smaller scale relative to the larger weights, their contribution to the NTK remains negligible. This property could be beneficial if the weights are later rescaled, such as during fine-tuning, potentially enhancing generalisation and transfer learning, as we will demonstrate in Sec. 3.5.3. We contrast this to the scenario where both weights are initialised with large Gaussian values, leading to *lazy* learning that maintains a fixed NTK but lacks any structural representation, as illustrated in Fig. 3.4D. Furthermore, in the infinite-width regime, where weights are initialised from a Gaussian distribution with large variance, averaging effects cause both input and output representations to approximate identity matrices. In this scenario, the network learns with minimal parameter variation, operating in the lazy regime with a fixed Neural Tangent Kernel (NTK). This behaviour contrasts with the dynamics observed in the current setting since both input and output representations are task agnostic – given by a scalled identity matrix.

Consequently, we propose a new *lazy* regime, which we refer to as the *semi-structured lazy* regime. These existing regimes preserve only the input or output representation, resulting in a partial loss of structural information. Altogether, we find that initialisation will determine which layer in the network the task specification features resides in: layers initialised with large values will be task-agnostic, while those initialised with small values will be task-specific.

3.4.3 Representation robustness and sensitivity to noise

Here, we examine the relationship between the learning regime and the robustness of the learned representations to added noise in the inputs and parameters. The expected post-convergence loss with added noise to the inputs is determined by the norm of the network function [40], which in our setting is independent of λ. Specifically, if we add zero-centered noise $\psi_{\mathbf{X}}$ with variance $\sigma_{\mathbf{X}}^2$ to the inputs, then the expected loss is $\langle \mathcal{L} \rangle_{\psi_{\mathbf{X}}} = \sigma_{\mathbf{X}}^2 \sum_{i=1}^{N_h} \tilde{\mathbf{S}}_i^2 + c$, where c is a constant that depends solely on the statistics of the training data (Figure 3.4E, Appendix A.3.3). However, if we instead add noise to the parameters, the expected loss scales quadratically with the norm of the weight matrices [40], which in our setting depend on λ. In particular, zero-centered parameter noise $\psi_{\mathbf{W}_1}$ and $\psi_{\mathbf{W}_2}$ with variance $\sigma_{\mathbf{W}}^2$ results in an expected loss of$\langle \mathcal{L} \rangle_{\psi_{\mathbf{W}_1}, \psi_{\mathbf{W}_2}} = \frac{1}{2} N_i \sigma_{\mathbf{W}}^2 ||\mathbf{W}_2||_F^2 + \frac{1}{2} N_o \sigma_{\mathbf{W}}^2 ||\mathbf{W}_1||_F^2 + \frac{1}{2} N_i N_h N_o \sigma^4 + c$, with norms $||\mathbf{W}_1||_F^2 = \frac{1}{2} \sum_{i=1}^{N_h} \left(\sqrt{4\tilde{\mathbf{S}}_i^2 + \lambda^2} + \lambda \right)$ and $||\mathbf{W}_2||_F^2 = \frac{1}{2} \sum_{i=1}^{N_h} \left(\sqrt{4\tilde{\mathbf{S}}_i^2 + \lambda^2} - \lambda \right)$. This implies that, under the assumption of equal input-output dimensions, networks initialised with $\lambda = 0$, corresponding to the rich regime, converge to solutions most robust to parameter noise (Fig. 3.4E, Appendix A.3.3). As the norm of λ increases, the loss scales proportionally, leading to less robust solutions. In practical terms, parameter noise can be interpreted as noise arising within the neurons of a biological network. In this context, rich solutions may offer more robust representations.

In conclusion, the impact of noise on the network is strongly influenced by both its source and the degree of weight balancing. While the impact of input noise is largely independent of the network's relative scale λ, the influence of parameter noise is shaped by the value of λ. Our findings further show that the rich regime consistently yields more robust solutions across a wide range of conditions.

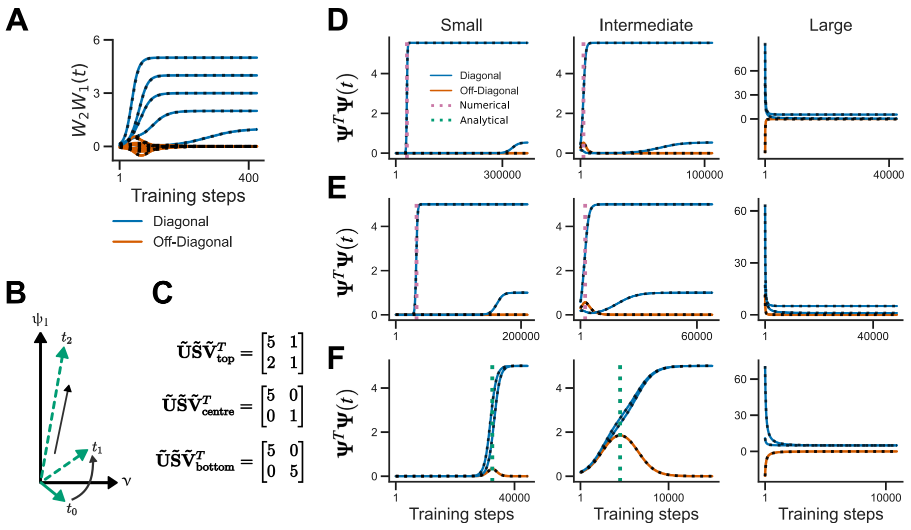


Figure 3.5: Decoupling dynamics. **A** Analytical (black dotted lines) and numerical (solid lines) of the temporal dynamics of the on- and off-diagonal elements of $\mathbf{\Psi}^T\mathbf{\Psi}$ in blue and red, respectively. **B** Schematic representation of the decoupling process. **C** Three target matrices with dense, unequal diagonal, and equal diagonal structure. **D-F** Decoupling dynamics for the top (D), middle (E), and bottom (F) tasks depicted in panel C. Row F contains analytical predictions for the time of the peak of the off-diagonal (dashed green). The network is initialised as defined in A.3.4 with small, intermediate and large variance. For simulation details, see Appendix A.5.7.

3.4.4 Dynamics of the singular vectors

We now seek to explore the dynamics of singular vectors in greater detail to gain a deeper understanding of the learning process. As highlighted in previous studies, learning is often driven by two interrelated processes that sometimes occur simultaneously: the growth of norms and the alignment of the basis [131].

Earlier analyses of learning dynamics were conducted under the assumption that the initial network weights are ”decoupled,” meaning the network's initial state and the task share the same singular vectors—that is, $\mathbf{U} = \tilde{\mathbf{U}}$ and $\mathbf{V} = \tilde{\mathbf{V}}$ [243]. Intuitively, this assumption implies that there is no cross-coupling between different singular modes, allowing each to evolve independently. In other words, the network's initial state was presumed to already encode part of the task's structure before training commenced, making it inherently aligned (see Sec. 3.4.1). However, this assumption rarely holds in real-world scenarios, where such pre-alignment is generally absent. As a consequence, most prior work has relied on the empirical observation that learning from *tabula rasa* small initial weights occurs in two phases: First, the network's input-output map rapidly decouples; then subsequently, independent singular modes are learned in this decoupled regime. Because this decoupling process is fast from small initial weights, the learning dynamics are still approximately described by the temporal learning dynamics of the singular values assuming decoupling from the start. This dynamic has been called a *silent alignment* process [18]. Here, we leverage our matrix Riccati approach to analytically study the dynamics of this decoupling process. We begin by deriving an alternate form of the exact solution that eases the analysis. For simplicity, we study the case where $\lambda = 0$

Theorem 3.4.4. *Let the weight matrices of a two-layer linear network be initialised by* $\mathbf{W}_1 = \mathbf{\Psi}(0)\tilde{\mathbf{V}}^T$ *and* $\mathbf{W}_2 = \tilde{\mathbf{U}}\mathbf{\Psi}(0)^T$*, where* $\mathbf{\Psi} \in \mathbb{R}^{N_h \times N_i}$ *is an arbitrary, invertible matrix. Then, under the assumptions of equal input-output dimensions, whitened inputs (1), zero-balanced weights (4) and full rank (described in section 3.3.5), the temporal dynamics of* $\mathbf{Q}\mathbf{Q}^T$ *are fully determined by*

$$\mathbf{\Psi}^T\mathbf{\Psi} = \left[e^{-\tilde{\mathbf{S}}\frac{t}{\tau}} \left(\mathbf{\Psi}(0)^T\mathbf{\Psi}(0)\right)^{-1} e^{-\tilde{\mathbf{S}}\frac{t}{\tau}} + (\mathbf{I} - e^{-2\tilde{\mathbf{S}}\frac{t}{\tau}})\tilde{\mathbf{S}}^{-1}\right]^{-1}. \tag{3.24}$$

For a proof of Theorem 3.4.4, please refer to Appendix A.3.4. We remark that this form is less general than that in Theorem 3.3.3, and in particular implies $\mathbf{U}\mathbf{V} = \tilde{\mathbf{U}}\tilde{\mathbf{V}}$. Here, the matrix $\mathbf{\Psi}^T\mathbf{\Psi}$ represents the dynamics directly in the SVD basis of the task. Off-diagonal elements represent counterproductive coupling between different singular modes (for instance, $[\mathbf{\Psi}^T\mathbf{\Psi}]_{21}$ is the strength of connection from input singular vector 1 to output singular vector 2, which must approach zero to perform the task perfectly), while on-diagonal elements represent the coupling within the same mode. For instance, $[\mathbf{\Psi}^T\mathbf{\Psi}]_{11}$ is the strength of connection from input singular vector 1 to output singular vector 1, which must approach the associated task singular value to perform the task perfectly. Hence, we can study the decoupling process by examining the dynamics by which $\mathbf{\Psi}^T\mathbf{\Psi}$ becomes approximately diagonal.

The outer inverse in Equation 3.24 renders it difficult to study high-dimensional networks analytically. Therefore, we focus on small networks with input and output dimension $N_i = 2$ and $N_o = 2$, for which a lengthy but explicit analytical solution is given in Appendix A.3.4. In this setting, the structure of the weight initialisation and task are encoded in the matrices

$$\mathbf{\Psi}(0)^T\mathbf{\Psi}(0) = \begin{bmatrix} \psi_1(0) & \nu(0) \\ \nu(0) & \psi_2(0) \end{bmatrix} \quad \text{and} \quad \tilde{\mathbf{S}} = \begin{bmatrix} \tilde{s}_1 & 0 \\ 0 & \tilde{s}_2 \end{bmatrix}, \tag{3.25}$$

where the parameters $\psi_1(0)$ and $\psi_2(0)$ represent the component of the initialisation that is aligned with the task, and $\nu(0)$ represents cross-coupling, such that taking $\nu(0) = 0$ recovers previously known and more restricted solutions for the decoupled case [243]. We use this setting to demonstrate two features of the learning dynamics.

3.4.4.1 Decoupling dynamics

We demonstrate that the learning dynamics can be decomposed into two distinct phases: a fitting phase, characterised by the growth of singular values, and an alignment phase, quantified by the decoupling of the singular vectors. We track decoupling by considering the dynamics of the off-diagonal element $\nu(t)$ (Fig. 3.5D-F red lines). At convergence, the off-diagonal element shrinks to zero, as shown in Appendix A.3.4. However, strikingly, $\nu(t)$ can exhibit non-monotonic trajectories with transient peaks or valleys partway through the learning process. In particular, in Appendix A.3.4 we derive the time of the peak magnitude as $t_{peak} = \frac{\tau}{4\tilde{s}} \ln \frac{\tilde{s}(\tilde{s}-\psi_1-\psi_2)}{\psi_1\psi_2-\nu(0)^2}$ (Fig. 3.5F green dotted line), which coincides approximately with the time at which the on-diagonal element is half learned. If initialised from small random weights, the off-diagonal remains near zero throughout learning, reminiscent of the silent alignment effect [18]. For large initialisations, no peak is observed and the dynamics are exponential. At intermediate initialisations, the maximum of the off-diagonal is reached before the singular mode is fully learned (Appendix A.3.4). Intuitively, a particular input singular vector can initially project appreciably onto the wrong output singular vector, corresponding to initial misalignment. This is only revealed when this link is amplified, at which point corrective dynamics remove the counterproductive coupling, as schematised in Fig. 3.5B. Taken together, our findings highlight the rich spectrum of learning dynamics that can be broadly characterised by two distinct phases: a fitting phase, marked by the growth of singular values, and an alignment phase, captured through decoupling. We recover the 'silent alignment' phenomenon, where rapid alignment precedes singular value growth, suggesting a clear separation between the two phases. In contrast, we also observe scenarios where these phases unfold concurrently, leading to altered singular value dynamics. Crucially, we show that the manifestation of these learning profiles depends sensitively on both the data and the initialisation. We leave the investigation of disentanglement behaviour under different values of λ for future work. Nonetheless, we expect to observe similar phenomenology as previously reported—specifically, the silent alignment effect. We report further measurements of decoupling in Appendix A.3.4.

3.4.4.2 Effect of initialisation variance

Next, we revisit the impact of initialisation scale for the on-diagonal dynamics. As shown in Fig. 3.5D-F, as the initialisation variance grows, the learning dynamics change from sigmoidal to exponential, possibly displaying more complex behaviour at intermediate variance (Appendix A.3.4). In this simple setting we can analyse this transition in detail. Taking $\tilde{s}_1 = \tilde{s}_2 = \tilde{s}$ as in Fig. 3.5F and $|\psi_1(0)|, |\psi_2(0)|, |\nu(0)| \ll 1$, we recover a sigmoidal trajectory,

$$\psi_1(t) = \frac{\tilde{s}\psi_1(0)}{e^{\frac{-2\tilde{s}t}{\tau}}\left[\tilde{s} - \psi_1(0) - \psi_2(0)\right] + \psi_1(0) + \psi_2(0)}, \tag{3.26}$$

while for $|\psi_1(0)|, |\psi_2(0)|, |\nu(0)| \gg 0$ the dynamics of the on-diagonal element ψ_1 is close to exponential (Fig. 3.5D-F left and right columns). The effect of initialisation variance (scale) across different lambda value is further analysed in Section 3.4.6.1.

3.4.5 The impact of the architecture

In this study, we examine how a network's learning regime is influenced by the interplay between its architecture and the sign of the *relative scale*. We analyse five types of network architectures, including three two-layer feedforward configurations illustrated in Fig. 3.6A: (1) *funnel networks*, which decrease in width from input to output ($N_i > N_h = N_o$); (2) *inverted-funnel networks*, which increase in width from input to output ($N_i = N_h < N_o$); and (3) *square networks*, where input, hidden, and output dimensions are equal ($N_i = N_h = N_o$). Additionally, we explore recurrent architectures.

3.4.5.1 Mirror flow

We analyse the effect of initialisation in feedforward architectures using a novel analytical frameworks that do not rely on Assumption (1) concerning whitened inputs. These analysis further enables the exploration of new architectural variants, including funnel networks and anti-funnel network structures, while also relaxing assumptions on the data structure.

We consider the same two-layer linear network setting used above with $N_h \geq \min(N_i, N_o)$, such that this parametrisation can represent all linear maps from $\mathbb{R}^{N_i} \to \mathbb{R}^{N_o}$. The rescaling symmetry between $\mathbf{W}_2$ and $\mathbf{W}_1$ implies the $N_h \times N_h$ matrix $\mathbf{\Lambda} = \eta_{w_2}\mathbf{W}_2(0)^\intercal\mathbf{W}_2(0) - \eta_{w_1}\mathbf{W}_1(0)\mathbf{W}_1(0)^\intercal$ is conserved throughout gradient flow [74].

We consider the dynamics of the network function $\boldsymbol{\beta} = \mathbf{W}_1^\intercal\mathbf{W}_2^\intercal \in \mathbb{R}^{N_i \times N_o}$,

$$\mathrm{vec}\left(\dot{\boldsymbol{\beta}}\right) = -\underbrace{\left(\eta_{w_2}\mathbf{W}_2(0)\mathbf{W}_2(0)^\intercal \oplus \eta w_1\mathbf{W}_1(0)^\intercal\mathbf{W}_1(0)\right)}_{\mathbf{M}}\mathrm{vec}(\mathbf{X}\mathbf{X}^\intercal\boldsymbol{\beta} - \mathbf{X}\mathbf{Y}^\intercal), \tag{3.27}$$

where $\mathrm{vec}(\cdot)$ denotes the vectorisation operator and $\oplus$ denotes the Kronecker sum[2].

We find that the dynamics of $\boldsymbol{\beta}$ are preconditioned by a matrix $\mathbf{M}$ that depends on quadratics of $\mathbf{W}_2$ and $\mathbf{W}_1$ and characterises the NTK matrix $\boldsymbol{NTK} = (\mathbf{I}_c \otimes \mathbf{X}^\intercal)\,\mathbf{M}\,(\mathbf{I}_c \otimes \mathbf{X})$. We now show how $\mathbf{M}$ can be expressed in terms of the rank-1 matrices $\boldsymbol{\beta}_k = \mathbf{W}_{1k}^\intercal\mathbf{W}_{2k}^\intercal \in \mathbb{R}^{N_i \times N_o}$, which represent the contribution to $\boldsymbol{\beta}$ of a single neuron with parameters $\mathbf{W}_{1k}$, $\mathbf{W}_{2k}$ and conserved quantity $\lambda_k = \mathbf{\Lambda}_{kk}$.

[2]The Kronecker sum is defined for square matrices $\mathbf{C} \in \mathbb{R}^{c \times c}$ and $\mathbf{D} \in \mathbb{R}^{d \times d}$ as $\mathbf{C} \oplus \mathbf{D} = \mathbf{C} \otimes \mathbf{I}_d + \mathbf{I}_c \otimes \mathbf{D}$.

Theorem 3.4.5. *Whenever* $||\boldsymbol{\beta}_k||_F \neq 0$ *for all* $k \in [N_h]$, *the matrix* $\mathbf{M}$ *can be expressed as the sum* $\mathbf{M} = \sum_{k=1}^{h} \mathbf{M}_k$ *over hidden neurons where* M_k *is defined as,*

$$\mathbf{M}_k = \left(\frac{\sqrt{\lambda_k^2 + 4\eta_{w_1}\eta_{w_2}||\boldsymbol{\beta}_k||_F^2} + \lambda_k}{2} \right) \frac{\boldsymbol{\beta}_k^\intercal \boldsymbol{\beta}_k}{||\boldsymbol{\beta}_k||_F^2} \oplus \left(\frac{\sqrt{\lambda_k^2 + 4\eta_{w_1}\eta_{w_2}||\boldsymbol{\beta}_k||_F^2} - \lambda_k}{2} \right) \frac{\boldsymbol{\beta}_k \boldsymbol{\beta}_k^\intercal}{||\boldsymbol{\beta}_k||_F^2}. \tag{3.28}$$

The proof of Theorem 3.4.5 can be found in Appendix A.3.5.2

In Appendix A.3.5.2, we derive $\mathbf{M}$ in the complementary setting where the network consists of a single neuron, i.e., $N_h = 1$. By studying the dependence of $\mathbf{M}$ on the conserved quantities λ_k and the dimensions N_i, N_h and N_o, we can determine the influence of the relative scale on the learning regime. Note that the first term in the equation corresponds to the update in $\mathbf{W_2}$, while the second term reflects the update in $\mathbf{W_1}$. The sign of λ therefore determines which layer predominantly adapts during training—positive λ emphasises adaptation in $\mathbf{W_2}$, whereas negative λ favours adaptation in $\mathbf{W_1}$. When $\min(N_i, N_o) \leq N_h < \max(N_i, N_h)$, and assuming independent initialisations for all β_k, then networks which narrow from input to output ($N_i > N_o$) enter the lazy regime when all $\lambda_k \gg 0$, whereas networks which expand from input to output ($N_i < N_o$) do so when all $\lambda_k \ll 0$. However, with opposite signs for λ_k, and assuming all $\boldsymbol{\beta}_k(0) \not\propto \boldsymbol{\beta}_*$, these networks enter a *delayed rich regime*. As elaborated in A.3.5.2, this occurs because, in these regimes, a solution $\boldsymbol{\beta}_*$ does not exist within the space spanned by $\mathbf{M}$ at initialisation. Intuitively, since the rate of change of one weight matrix is proportional to the norm of the other, a strong imbalance—where one matrix is much larger than the other—can lead to the effective freezing of the larger matrix. In such cases, lazy learning requires that the target lies within the span of the initialisation matrix (i.e., the task is not in the null space of the network), a scenario typically associated with the lazy regime $\lambda_k \gg 0$ if $N_i > N_o$ and $\lambda_k \ll 0$ if $N_i < N_o$) and $\lambda_k \gg 0$. If this condition is not satisfied—i.e., respectively $\lambda_k \ll 0$ is small or negative—the network enters a regime where the smaller matrix must first grow in norm before meaningful alignment can occur. This explains the delayed onset of learning observed in the delayed rich regime. More details are provided in the Appendix A.3.5. When $N_h \geq \max(N_i, N_o)$ all networks enter the lazy regime with all $\lambda_k \gg 0$ or all $\lambda_k \ll 0$. When the network is square, it enters the lazy regime in the limit as $\lambda \to \pm\infty$, as first shown in Section 3.4. Conversely, as all $\lambda_k \to 0$, all networks transition into the rich regime regardless of dimensions.

The analytical framework established in Theorem 3.4.5 offers valuable insights into the learning regime under varying limits of λ_k across different architectural configurations. While this analytical method does not capture the exact dynamics-which we develop further below in Sec. 3.4.5.2 - it reveals that the expressivity of the network is primarily governed by the layer with the larger input or output dimensionality. When the layer that predominantly adapts during training does not correspond to the more expressive (i.e., wider) layer, the network may exhibit a phenomenon referred to as *delayed richness*. These findings provide a more nuanced perspective on the results described above for square networks and aligned networks. Importantly, within this analytical framework, we do not require Assumption 1 concerning whitened inputs or isotropic initialisation (Assumption 2: $\mathbf{\Lambda} = \lambda \mathbf{I}_{N_h}$), yet we recover the findings outlined previously in Sec. 3.4, where the magnitude of the relative scale parameter λ governs the transition between rich and lazy learning regimes. Specifically, we further demonstrate that the relationship between the limiting behaviour of λ and the resulting learning regime is architecture-dependent.

While Theorem 3.4.5 offers valuable insight into the learning regimes in the limits of λ_k, understanding the transition between regimes remains challenging. To achieve this, we aim to express $\mathbf{M}$ directly in terms of $\boldsymbol{\beta}$, rather than the layer-specific parameters $\boldsymbol{\beta}_k$, by imposing structure on the conserved quantities $\mathbf{\Lambda}$.

Theorem 3.4.6. *When $\mathbf{\Lambda} = \lambda \mathbf{I}_{N_h}$ and $N_h = N_i$ if $\lambda < 0$ or $N_h = N_o$ if $\lambda > 0$, then the matrix M can be expressed as $\mathbf{M} = \sqrt{\eta_{w_1}\eta_{w_2}\boldsymbol{\beta}^\intercal\boldsymbol{\beta} + \frac{\lambda^2}{4}\mathbf{I}_{N_o}} \otimes \mathbf{I}_{N_i} + \mathbf{I}_{N_o} \otimes \sqrt{\eta_{w_2}\eta_{w_1}\boldsymbol{\beta}\boldsymbol{\beta}^\intercal + \frac{\lambda^2}{4}\mathbf{I}_{N_i}}$. The proof is provided in Appendix A.3.5.3.*

From Theorem 3.4.6 the resulting dynamics of $\boldsymbol{\beta}$ simplify to a self-consistent equation regulated by λ,

$$\dot{\boldsymbol{\beta}} = -\mathbf{X}\mathbf{P}\sqrt{\eta_a\eta_w\boldsymbol{\beta}^\intercal\boldsymbol{\beta} + \frac{\lambda^2}{4}\mathbf{I}_{N_o}} - \sqrt{\eta_a\eta_w\boldsymbol{\beta}\boldsymbol{\beta}^\intercal + \frac{\lambda^2}{4}\mathbf{I}_{N_i}}\mathbf{X}\mathbf{P}, \tag{3.29}$$

where $\mathbf{P} = \mathbf{X}^\intercal\boldsymbol{\beta} - Y$ is the residual. Under our isotropic assumption on the conserved quantities $\Lambda = \lambda\mathbf{I}_{N_h}$, these dynamics are exact. Concurrent to our work, Tu et al. [280] finds that $\boldsymbol{\beta}$ *approximately* follows these dynamics in the overparameterised setting $N_h \gg \max(N_i, N_o)$ under a Gaussian initialisation $\mathcal{N}(0, \sigma^2)$ of the parameters where $\sigma^2 N_h$ is analogous to λ. By studying the dependence of $\mathbf{M}$ on the conserved quantities λ, we can determine the influence of the relative scale on the learning regime. We arrive at similar conclusions to those described above, while taking into consideration that we are analysing a more restricted architectural setting, as outlined in Appendix A.3.5.3.

Next we attempt to better understand these dynamics for intermediate values of λ through the lens of a mirror flow. A brief review of the mirror flow analysis is provided in the Appendix A.3.5.1. Equipped with a self-consistent equation for the dynamics of $\boldsymbol{\beta}$, we now aim to interpret these dynamics as a mirror flow with a λ-dependent potential.

Theorem 3.4.7. *Let $\boldsymbol{\Lambda} = \lambda\mathbf{I}_{N_h}$ and assume $N_h \geq \min(N_i, N_o)$ and $\mathbf{S} \neq 0$. We then have that the dynamics of $\mathbf{S}$, the singular values of $\boldsymbol{\beta}$, are given by the mirror flow*

$$\dot{\mathbf{S}} = -\left(\nabla^2\Phi_\lambda(\mathbf{S})\right)^{-1}\nabla_{\mathbf{S}}\mathcal{L}, \tag{3.30}$$

where $\mathcal{L}$ is the loss, $\Phi_\lambda(\mathbf{S}) = \sum_{i=1}^{\min(N_i,N_o)} q_\lambda(\mathbf{S}_i)$ and q_λ is the hyperbolic entropy potential

$$q_\lambda(x) = \frac{1}{4}\left(2x\sinh^{-1}\left(\frac{2x}{|\lambda|}\right) - \sqrt{4x^2+\lambda^2} + |\lambda|\right). \tag{3.31}$$

The proof of the Theorem can be found in Appendix A.3.5.4. Theorem 3.4.7 implies that the dynamics for the singular values of $\boldsymbol{\beta}$ can be described as a mirror flow with a λ-dependent potential. This potential was first identified as the inductive bias for diagonal linear networks [294], and the same mirror flow on the singular values is derived from a different initialisation choice in prior work [285]. Termed *hyperbolic entropy*, this potential smoothly interpolates between an ℓ^1 and ℓ^2 penalty on the singular values for the rich ($\lambda \to 0$) and lazy ($\lambda \to \pm\infty$) regimes respectively. Unfortunately, in our setting we cannot adapt our mirror flow interpretation into a statement on the inductive bias at interpolation because the singular vectors evolve through training. If we introduce additional assumptions — specifically, whitened input data ($\mathbf{X}^\intercal\mathbf{X}^\intercal = \mathbf{I}_{N_i}$) and a task-aligned initialisation such that the singular vectors of $\boldsymbol{\beta}_0$ are aligned with those of $\boldsymbol{\beta}_*$ — we can ensure that the singular vectors remain constant and thus derive an inductive bias on the singular values. However, in this setting the dynamics decouple completely, implying there is no difference between applying an ℓ^1 or ℓ^2 penalty on the singular values. Consequently, even though the dynamics will depend on λ, the final interpolating solution will be independent of λ, making a statement on the inductive bias insignificant.

All together, the methods presented above further enables the exploration of new architectural variants, including funnel networks and anti-funnel network, while also relaxing assumptions on the data structure (ie.whitened inputs).

3.4.5.2 Feed-forward architecture

Now, we analyse the influence of architecture using our exact solution, $\mathbf{QQ}^T$, which characterises the dynamics of the NTK across various feedforward network architectures. To examine the NTK's evolution under varying λ initialisations, we compute the kernel distance from initialisation, as defined in Fort et al. [91].

As shown in Fig. 3.6B, we observe that funnel networks consistently enter the *lazy* regime as $\lambda \to \infty$, while inverted-funnel networks do so as $\lambda \to -\infty$. The NTK remains static during the initial phase, rigorously confirming the rank argument first introduced above for the multi-output setting. In the opposite limits of λ, these networks transition from a *lazy* regime to a *rich* regime. During this second

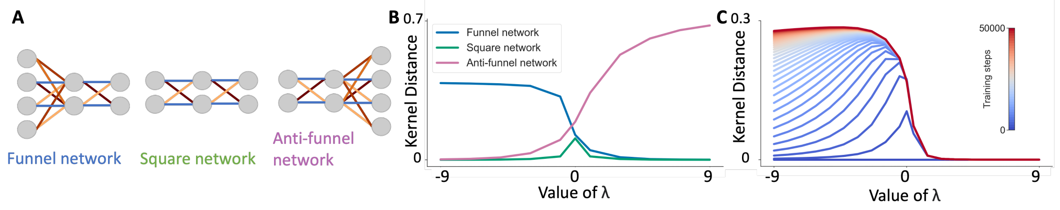


Figure 3.6: **A.** Schematic representations of the network architectures considered, from left to right: funnel network, square network, and inverted-funnel network. **B.** The plot shows the NTK kernel distance from initialisation, as defined in [91] across the three architectures depicted schematically. **C.** The NTK kernel distance away from initialisation over training time. For simulation details, see Appendix A.5.8.

alignment phase, the NTK matrix undergoes changes, indicating an initial *lazy* phase followed by a *delayed rich* phase. We further investigate and quantify this *delayed rich* regime, showing the NTK movement over training in Fig. 3.6C. This behaviour is also quantified in Theorem A.3.13, which describes the rate of learning in this network.

Intuitively, the delayed onset of the rich regime occurs because no least-squares solution exists within the span of the network at initialisation. In such cases, the network enters a delayed rich phase, where λ tends toward infinity, with the magnitude of λ determining the length of the delay. At first, the network exhibits *lazy* dynamics, striving to approximate the solution. However, as constraints necessitate adjustments in its directions, the network gradually transitions into the *rich* phase. Interestingly, delayed rich regimes have been suggested as the mechanism underlying grokking—where networks trained on tasks suddenly exhibit strong generalisation long after memorising the training data [155].

For square networks with equal input and output dimensions, this behaviour is discussed in Sec. 3.4.2. Across all architectures, as $\lambda \to 0$, the networks consistently transition into the *rich* regime. Altogether, we further characterise the *delayed rich* regime in wide networks.

3.4.5.3 Recurrence facilitates rich learning

We have identified two distinct learning regimes in artificial neural networks: feature-learning (*rich* learning) and non-feature-learning (*lazy* learning), both of which are influenced by weight initialisation in deep feedforward architectures [12, 21, 39, 119, 300, 87, 69, 67]. However, theoretical investigations into how non-feedforward architectures affect these learning regimes remain limited [81]. Notably, Liu et al. [176] examined the role of weight connectivity in shaping learning regimes in RNNs. Building on this line of inquiry, we explore how recurrence impacts feature-learning dynamics. Specifically, we investigate whether recurrent architectures impose additional constraints on the learning problem, thereby biasing the network towards the rich learning regime.

We study a LRNN (Fig. 3.7) parametrised by matrices $\mathbf{W}_1 \in \mathbb{R}^{N_h \times N_i}, \mathbf{W}_h \in \mathbb{R}^{N_h \times N_h}, \mathbf{W}_2 \in \mathbb{R}^{N_o \times N_h}$ with a hidden state $\mathbf{h}_t \in \mathbb{R}^{N_h}$ that receives an input $\mathbf{x}_t \in \mathbb{R}^{N_x}$ at each timestep t and updates its hidden state. For simplicity, we study the *single-output* case, where the network only produces an output $\hat{\mathbf{y}}_T \in \mathbb{R}^{N_o}$ at the last timestep T. The network is characterised by the equations

$$\mathbf{h}_{t+1} = \mathbf{W}_h \mathbf{h}_t + \mathbf{W}_1 \mathbf{x}_t \; , \tag{3.32}$$

$$\hat{\mathbf{y}}_T = \mathbf{W}_2 \mathbf{h}_{T+1} \; . \tag{3.33}$$

We initialise the hidden layer $\mathbf{h}_1$ as a vector of zeros, yielding

$$\mathbf{h}_{t+1} = \sum_{i=1}^{t} \mathbf{W}_h^{t-i} \mathbf{W}_1 \mathbf{x}_i \; . \tag{3.34}$$

A schematic of the networks can be found in Fig. 3.7 We analyse learning in the LRNN when trained using backpropagation through time on the squared error over P trajectories $\{\mathbf{x}_{p,1}, \mathbf{x}_{n,2}, \ldots, \mathbf{x}_{p,T}, \mathbf{y}_{p,T}\}_{n=1}^{P}$

$$\mathcal{L} = \frac{1}{2} \sum_{n=1}^{P} ||\mathbf{y}_{n,T} - \mathbf{W}_2 (\sum_{i=1}^{T} \mathbf{W}_h^{T-i} \mathbf{W}_1 \mathbf{x}_{n,i})||^2 \tag{3.35}$$

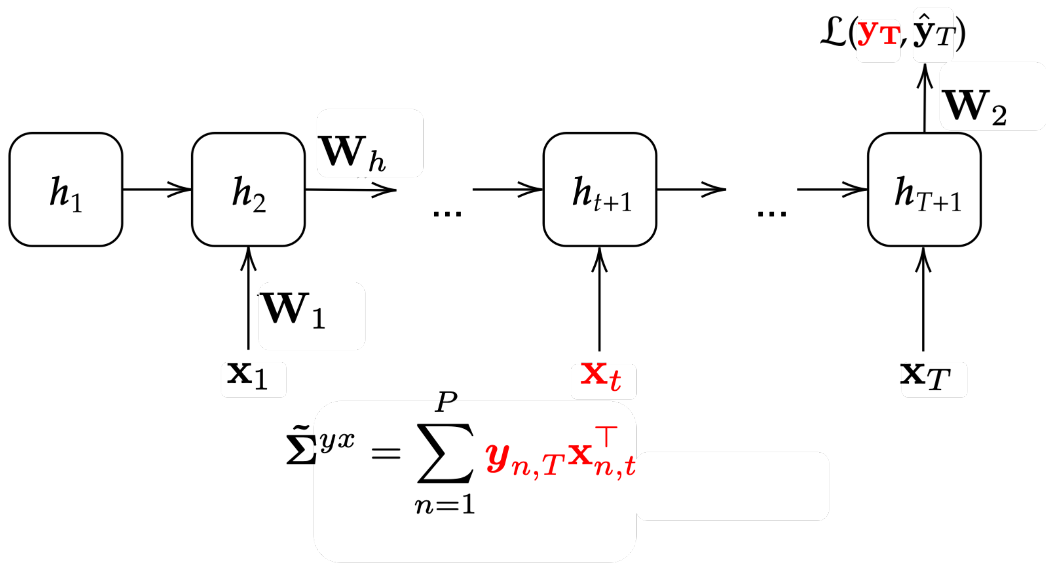

$\mathcal{L}(\mathbf{y_T}, \hat{\mathbf{y}}_T)$
$\mathbf{W}_2$
h_1
h_2
$\mathbf{W}_h$
h_{t+1}
h_{T+1}
$\mathbf{W}_1$
$\mathbf{x}_1$
$\mathbf{x}_t$
$\mathbf{x}_T$
$\tilde{\mathbf{\Sigma}}^{yx} = \sum_{n=1}^{P} \mathbf{y}_{n,T}\mathbf{x}_{n,t}^\top$

Figure 3.7: Linear RNN model captures task dynamics through temporally-dependent singular values. The data correlation matrices Σ^{YX_t} have constant left and right singular vectors, varying only in their singular values S_t across time. For details, see Appendix A.5.9.

With the model and loss function fixed, our next step is to specify a task for the model to learn. In this linear setting, the task is fully specified by the sequence of matrices $\tilde{\mathbf{\Sigma}}^{yx} = \sum_{n=1}^{P} \mathbf{y}_{n,T}\mathbf{x}_{n,t}^\top$, the input-output correlation matrix between the input $\mathbf{x}_{n,t}$ at timestep t and the final output $\mathbf{y}_{n,T}$.

We derive a general form of the NTK for this setting (i.e. finite-width LRNNs) in Appendix A.3.5.6, extending the analysis of Emami et al. [81], which focused on the infinite-width limit. We then leverage this formulation to investigate the learning regimes that arise in LRNNs under varying initialisations and trajectory lengths. Note that in our setting, trajectory lengths correspond to the depth of the network. We examine two initialisation settings: one where the weights are misaligned (Fig. 3.8(left)) and another where they are aligned (Fig. 3.8 (right)). To quantify feature learning, we measure the kernel distance between the NTK at initialisation and the end of training for LRNNs as a function of trajectory length and weight initialisation scale. We train on constant task dynamics in both the aligned and unaligned cases. Using networks without aligned weights also broadens the possible solution space, which is important when considering, for example, how

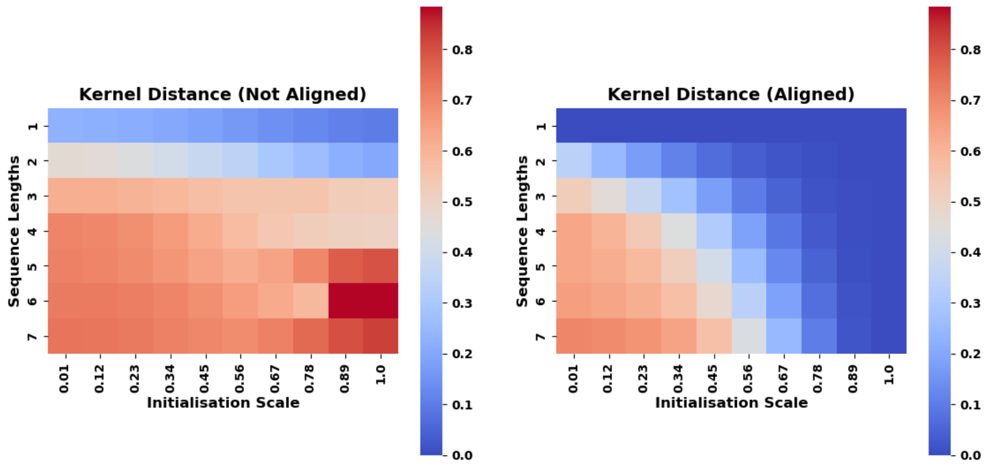


Figure 3.8: Recurrence drives feature learning. Phase plots illustrating the kernel distance of the NTK from initialisation as a function of trajectory length and initialisation scale for LRNN initialised with weights that are (*left*) unaligned and (*right*) aligned. See experimental details in Appendix A.5.10

learning gives rise to different dynamic solutions in networks trained on the same tasks [281]. As expected, in the aligned case, we see that the kernel moves further in networks with smaller initialisations (rich learning) compared to networks with large weights (lazy learning) relative to the target. Under the aligned configuration, vector rotation to align with the task is unnecessary; instead, the scaling of the initialisation becomes the critical factor. When the network's initialisation variance is close to the target variance, the adjustments required to fit the target are minimised. This explains why smaller initialisation scales result in more pronounced NTK movement. More interestingly, we also find that the kernel distance increases as the network transitions from feedforward computations (i.e. trajectory lengths/depth 1 and 2) to a recurrent network with longer trajectory lengths (Fig.3.8), indicating recurrence induces greater feature learning for networks misaligned to the target at initialisation. The emergence of the rich regime in the presence of recurrence is not believed to share the same origin as that observed under balanced conditions; the underlying mechanisms are thought to be fundamentally different. In Appendix A.3.5.6, we further show that this effect only appears to exist in networks with finite width and that lazy learning becomes more prevalent in wider networks, as previously observed in feedforward networks [165]. This framework offers a systematic approach to understanding how recurrence, initialisation scale, and temporal structure influence the learning dynamics of neural networks. In the aligned setting, we use singular value decomposition (SVD) to analyse the network's learning regime. Building on this, Proca et al. [219] take a step further by providing exact solutions to the learning dynamics using the eigenvalue decomposition, which accommodates complex eigenvalues. This approach opens the door to analysing rotational and oscillatory behaviours—dynamics that are likely essential for understanding how recurrent neural networks (RNNs) learn and process information, especially in biologically inspired contexts. Further exploring these richer dynamics will be a key direction for future work. Additionally, understanding the role of relative scaling within this framework remains a significant and open challenge to be addressed in subsequent research.

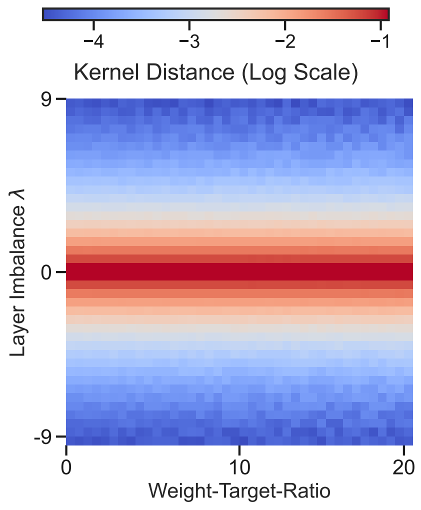
3.4.6 The impact of scale

Up to this point, we have analysed the influence of relative scale, λ, on the learning regime. However, absolute scale has traditionally been considered the primary factor driving the learning regime [243]. To further investigate this, we will independently vary both absolute and relative scale to understand their impact on the learning regime. Additionally, we will examine how the ratio of scale to target scale influences learning dynamics when the balanced condition is not exactly enforced.

3.4.6.1 Scale vs relative scale

A straightforward intuition for the scale and the relative scale can be gained by considering the scalar case where $N_i = N_h = N_o = 1$. In this scenario, it is easy to ensure that $w_1^2 = w_2^2$ satisfies $\lambda = 0$ while allowing for different absolute scales. For instance, $w_1 = w_2 = 0.001$ or $w_1 = w_2 = 5$. In such cases, the absolute scale is clearly decoupled from the relative scale. However, in more complex settings, the relative scale and absolute scale interact in non-trivial ways.

Previously, our primary focus was on how λ, the relative scale parameter, governs the transition between sigmoidal and exponential dynamical regimes. However, the role of the absolute scale is also evident in Theorem 3.4.1. A similar argument applies to the absolute scale, which explicitly appears as $s_\alpha(0)$ in these equations. When $\lambda = 0$, the dynamics of s_α simplify to the classical solution of the Bernoulli differential equation. In the limiting case where $s_\alpha(0) \to 0$, the system follows classic sigmoidal dynamics, characteristic of the rich regime. Conversely, as $s_\alpha(0) \to \infty$, the system transitions to exponential dynamics, characteristic of the lazy regime, as previously noted in Saxe et al. [243].



(a) Layer Imbalance

Figure 3.9: Two ways to get lazy dynamics. Figures show the changes in NTK (measure of non-linear dynamics) of linear neural network trained on randomised regression. **A**: In Sec. 3.4.6.1, when the weights are *lambda-balanced* (Assumption 2), layer imbalance governs the transition between rich (red) and lazy (blue) dynamics.

However, this influence is not explicitly apparent in the main theorem. The effect of absolute scale is inherently embedded within our framework through the definitions of **B**, **C**, and **D** (see Eq. 3.11). To further explore this relationship, we numerically investigate the interplay between relative weight scale, absolute weight scale, and the network's learning dynamics in a general setting. To do so, we measure the kernel distance between the NTK at initialisation and at the end of training for networks initialised with random weights of specified relative and absolute scales. The networks are trained on a randomly generated input-output task. We define the absolute scale of the weights as the norm of $\mathbf{W}_2\mathbf{W}_1$. By keeping the target fixed at order one, we specifically examine the effect of the weight-to-target ratio. Fig. 3.9A illustrates that in a square ($N_i = N_o$) linear neural network, rich dynamics emerge when the weights are balanced with a small λ[3]. As introduced in Sec. 3.4.2, this indicates that there exist initial states with large zero-balanced weights that lead to rich solutions. Given this, key questions arise: What are the resulting system dynamics? Does the network continue to exhibit the stepwise evolution characteristic of the rich regime, or does it instead follow the exponential dynamics associated with large-scale initialisation?

To explore this, we again consider the semantic learning task that maps living entities to positions in a hierarchy (Fig. 3.10A,B) [243]. When training a two-layer network from small random initial weights, the input and output representational similarity matrices (RSM) (Fig. 3.10C, upper left and lower right quadrants) align with the task structure at convergence. However, when initialised with large weights, the RSM indicates that the network has instead converged to a *lazy* solution (Fig. 3.10D). Importantly, the final function learned by the network remains identical in both cases (Fig. 3.10C, D, lower left quadrant). Additionally, despite identical final loss, the learning dynamics differ significantly: small initial weights lead to a slow, stepwise evolution, whereas large initial weights result in rapid, exponential learning (Fig. 3.10F), as predicted by Saxe et al. [243]. We call this regime the *rapid rich* regime. Intuitively, the system enters the rapid rich regime due to two main factors. First, the update rate of the weights is proportional to their magnitude: the rate of change of $\mathbf{W_1}$ is modulated by the norm of $\mathbf{W_2}$, and vice versa. As a result, larger weights facilitate faster updates, creating a positive feedback loop that accelerates learning dynamics. Second, when $\lambda = 0$, the dynamics consistently drive the system into the rich regime, as confirmed by our results in A.3.4.

[3]In Appendix A.3.6.1, we further analyse the phase portrait of the learning regime across different architectures as a function of both scale and relative scale.

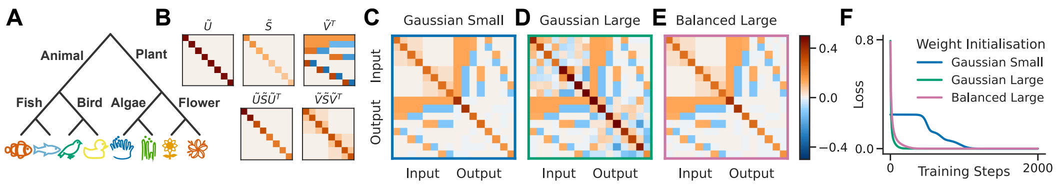


Figure 3.10: Rich and lazy learning. **A** Semantic learning task, **B** SVD of the input-output correlation of the task (top) and the respective RSMs (bottom). Rows and columns in the SVD and RSMs are identically ordered as the order of items in the hierarchical tree. **C** Final $\mathbf{QQ}^T$ matrices after training converged when initialised from random small weights, **D** random large weights (note how the upper left and lower right quadrant differ from the task's RSMs) and **E** large zero-balanced weights. **F** Learning curves for the three different initialisations as in C (green), D (pink) and E (blue). While both large weight initialisations lead to fast exponential learning curves, the small weight initialisation leads to a slow step-like decay of the loss. For details of the experiment, see Appendix A.5.12

Interestingly, with zero-balanced initialisation and large weights, certain initial states can lead to *rich* solutions while exhibiting fast exponential learning curves (Fig. 3.10E, F). We reveal a class of task-independent initialisations that radically alter learning dynamics from slow non-linear dynamics to fast exponential trajectories while converging to a global optimum with identical representational similarity, dissociating learning trajectories from the structure of initial internal representations. For additional simulations, see Appendix A.3.6.2. Thus, our framework effectively captures the transition from stepwise to exponential learning dynamics as the weight scale increases, distinguishing this effect from the structure of internal representations. This underscores the complex interplay between scale and relative scale in shaping the learning regime.

3.5 Applications

In this section, we apply the exact solutions for the learning dynamics in deep linear networks described in Sec. 3.3 to illustrate several phenomena relevant to machine learning and neuroscience.

3.5.1 Continual learning

A variety of theoretical work has investigated aspects of continual learning [17, 64, 167, 256, 279]. In this setting, starting from an initial set of weights, a network is trained on a sequence of tasks with respective input-output correlations $\mathcal{T}_1 = \tilde{\mathbf{\Sigma}}_1^{yx}, \mathcal{T}_2 = \tilde{\mathbf{\Sigma}}_2^{yx}, \ldots \mathcal{T}_n = \tilde{\mathbf{\Sigma}}_n^{yx}$. As illustrated in Fig. 3.11A, our framework enables exact analytical solutions for the entire continual learning process, specifically for the evolution of the network function $\mathbf{W_2W_1}$. In this setting, the final state of the network after training on one task serves as the initial state for the next. The dynamics observed across tasks show rich variability, with a clear distinction between learning from random initialisation and adaptation during inter-task transitions. At initialisation, small random weights produce sigmoidal learning dynamics, whereas post-task weights—having already reached a larger norm—induce more exponential-like learning dynamics in subsequent tasks.

These solutions also precisely characterise the temporal evolution of forgetting across arbitrary task sequences, as forgetting is entirely governed by the dynamics of the network function $\mathbf{W_2W_1}$. Training on later tasks can overwrite previously learned knowledge, a phenomenon known as catastrophic forgetting —quantified as the relative change in loss [193, 227, 92]. The pathology of catastrophic forgetting has long been a challenge for neural network models [193, 227, 92]. We offer additional theoretical insights into the underlying basis of this phenomenon. As detailed in Appendix A.4.1 (Fig. 3.11A), we demonstrate that, regardless of the chosen value of λ, training on subsequent tasks can result in the overwriting of previously acquired knowledge, leading to catastrophic forgetting [193, 227, 92].

As a result, catastrophic forgetting is entirely determined by the similarity of the input-output correlations between tasks. This implies that the amount of forgetting can be fully determined for a sequence of tasks before the onset of training. Specifically, task similarity is assessed via their shared singular value/vector structure. When tasks share common singular values/vectors, learning is facilitated, as the network can reuse previously acquired representational components without the need to relearn them. Using our analytical framework, we precisely describe the dynamics that give rise to catastrophic forgetting in linear networks. Importantly, we show that the forgetting profile is independent of initialisation — indicating that prior knowledge, in this context, does not mitigate forgetting. However, in regimes where early stopping is employed, the dynamics of forgetting become sensitive to both the order in which tasks are learned and the initial choice of the parameter λ. In such cases, our exact solutions can serve as a valuable tool for predicting the extent of forgetting and for informing training strategies to mitigate its impact. It is important to note that our theoretical model does not incorporate noise in the data — an important factor in early stopping scenarios where training halts before overfitting occurs. While we cautiously anticipate that the influence of λ on the rich and lazy regime dynamics will remain consistent under mild noise conditions, this remains an open question. Further investigation is required to confirm whether the analytical insights developed here extend robustly to noise.

As expected, our findings are specific to linear networks. In contrast, non-linear networks using activations such as tanh or ReLU exhibit different behaviours: their weights become rapidly unbalanced, and the forgetting values computed prior to training no longer accurately predict the actual outcome (see Fig.3.11B). The phenomenon of catastrophic forgetting in non-linear settings remains poorly understood. In the following section (Sec.3.5.1), we take further steps toward characterising the forgetting profile as a function of initialisation.

In summary, our results describe the exact learning dynamics of catastrophic

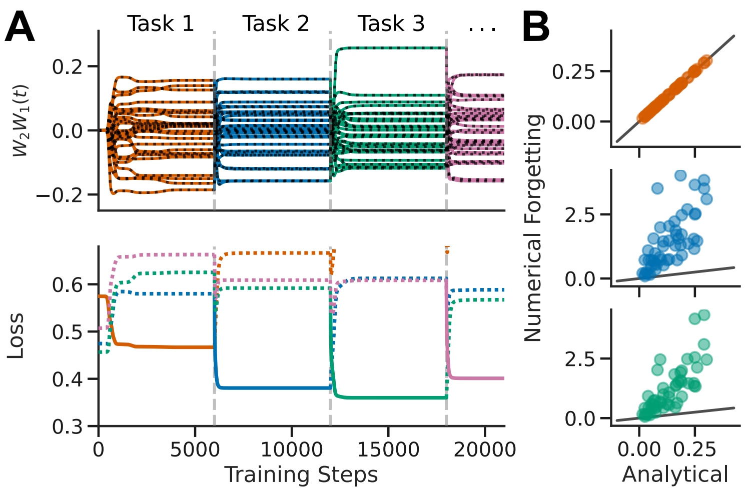


Figure 3.11: Continual learning. **A** Top: Network training from small zero-balanced weights on a sequence of tasks (coloured lines show simulation and black dotted lines analytical results). Bottom: Evaluation loss for tasks of the sequence (dotted) while training on the current task (solid). As the network function is optimised on the current task, the loss of other tasks increases ($\lambda = 0$). **B** Comparison of the numerical and analytical amount of catastrophic forgetting on a first task after training on a second task for $n = 50$ linear (red), tanh (blue) and ReLU (green) networks. Each dot correspond to a different initialisation seed. For simulation details, see Appendix A.5.13.

forgetting and thus provide an analytical tool to study its mechanisms and potential countermeasures.

3.5.2 Reversal learning

During reversal learning, previously acquired knowledge must be relearned, necessitating the overcoming of an earlier established relationship between inputs and outputs. For example, reversal learning occurs when items of a class are mislabelled and later corrected (Fig. 3.12C, D top). The challenge in such tasks lies in the persistence of the old association, which isn't simply erased but often actively competes with the formation of the new one. We show analytically that reversal learning, in fact, does not succeed in deep linear networks when $\lambda = 0$ (Appendix A.4.2). The pre-existing knowledge lies exactly on the separatrix of a saddle point, causing the learning dynamics to converge to zero (Fig. 3.12A). In contrast, the learning still succeeds numerically, as any noise will perturb the dynamics off the saddle point, allowing learning to proceed (Fig. 3.12A). However, the dynamics are slow in the vicinity of the saddle point, providing a theoretical explanation for catastrophic slowing in deep linear networks [168].

This scenario can be interpreted as the ongoing competition between memory traces: the established neural memory supporting the old behaviour (the old memory trace) creates a pull or a difficult landscape feature (the saddle point) that makes it hard for the network to efficiently forge the new memory. Therefore, our theoretical framework provides a potential mechanism for the competing memory trace hypothesis, a prominent idea in neuroscience [283]. Our results suggest that, rather than being passively overwritten, prior knowledge (the old memory trace) actively interferes with the formation of new memory by shaping the learning landscape, thereby hindering efficient adaptation.

However, when λ is non-zero, reversal learning dynamics consistently succeed, as they avoid passing through the saddle point due to the initialisation scheme. This is both theoretically proven and numerically illustrated in Appendix A.4.2. We also present a spectrum of reversal learning behaviours controlled by the *relative scale* λ, ranging from *rich* to *lazy* learning regimes. Further, the exact learning dynamics reveal that (Appendix A.4.2) shallow networks also succeed without exhibiting catastrophic slowing during reversal learning (Fig. 3.12B). Altogether, this spectrum of reversal behaviours has the potential to explain the diverse dynamics observed in animal behaviours, offering insights into the learning regimes relevant to various neuroscience experiments.

3.5.3 Transfer learning

We consider how different λ initialisations influence generalisation to a new feature after being trained on an initial task. As detailed in Appendix A.4.3, we first train each network on the hierarchical semantic learning task described in Fig. 3.10. We then add a new feature to the dataset, and train the network specifically on the corresponding item while keeping the rest of the network parameters unchanged. Afterwards, we evaluate the generalisation to the other items. We observe in Appendix

Fig. A.10 that the hierarchical structure of the data is effectively transferred to the new feature when the representation is task-specific and λ is zero. Conversely, when the input feature representation is *lazy* ($\lambda \leq 0$), meaning the hidden representation lacks adaptation, no hierarchical generalisation is observed. Strikingly, when λ is positive, the hierarchical structure in the input weights remains small but structured, while the output weights exhibit a *lazy* representation and the network generalises hierarchically. Specifically, Appendix Fig. A.10 shows that increasing λ leads to better generalisation. Therefore, as λ increases, networks more effectively transfer the hierarchical structure of the network to the new feature for untrained items, leading to an increase in generalisation performance. This indicates that the *lazy* regime structure (large λ values) first defined in Sec. 3.4 can be beneficial for transfer learning.

3.5.4 Fine-tuning

It is a common practice to pre-train neural networks on a large auxiliary task before fine-tuning them on a downstream task with limited samples. Despite the widespread use of this approach, the dynamics and outcomes of this method remain poorly understood. In our study, we provide a theoretical foundation for the empirical success of fine-tuning, aiming to improve our understanding of how performance depends on the initialisation. Specifically, we investigate how changes in λ-balancedness after pretraining—denoted as λ_{FT}—may affect fine-tuning performance on a new dataset, as discussed in Appendix A.4.4. Across all tasks we consider, we consistently find that fine-tuning performance improves and converges more quickly as networks are re-balanced to larger values of λ_{FT} and, conversely, decreases as λ_{FT} approaches zero as shown in Fig. A.11. Interestingly, networks that are not re-balanced before finetuning (i.e., with $\lambda_{\text{FT}} = \emptyset$) but were initialised with λ_{PT} [4] prior to pretraining, perform similarly on new tasks to networks that are re-balanced such that $\lambda_{\text{FT}} = \lambda_{\text{PT}}$. In this work, we analyse the fine-tuning dynamics of two-layer linear networks and show that rebalancing enables models to leverage both the bene-

[4]PT: Pre-training

fits and trade-offs of distinct learning regimes governed by λ. While conceptually simple, the two-layer linear architectures studied in this work are widely employed in practice for fine-tuning large pre-trained language and vision models. Notably, they serve as the foundation for methods such as Low-Rank Adapters (LoRA)[124], which are discussed in more detail in Appendix A.4.4. Although the specific setup considered here differs in some respects—most notably, the input-output mapping is not initialised to zero—these models exhibit comparable behaviours that make our findings broadly relevant. In particular, prior work has shown that imbalances in such systems can lead to both benefits and drawbacks in terms of fine-tuning stability and performance. While a comprehensive empirical investigation of how initialisation influences fine-tuning outcomes is beyond the scope of this thesis, it represents a compelling and important direction for future research.

3.5.5 Revising structured knowledge

Knowledge is often organised within an underlying, shared structure, of which many can be learned and represented in deep linear networks [247]. For example, spatial locations can be related to each other using the same cardinal directions, or varying semantic knowledge can be organised using the same hierarchical tree. Here, we investigate if deep linear networks benefit from shared underlying structure. To this end, a network, initialised with $\lambda = 0$ is first trained on the three-level hierarchical tree of Sec. 3.4 (eight items of the living kingdom, each with a set of eight associated features), and subsequently trained on a revised version of the hierarchy. When training the network on a new hierarchical tree with identical items but a new set of features, like a colour hierarchy (Fig. 3.12C, D bottom), there is no speed advantage in comparison to a random initialisation with similar initial variance (Fig. 3.12E-F, bottom). Importantly, from Theorem A.3.3, it follows that the learning process can be sped up significantly by initialising from large zero-balanced weights while converging to a global minimum with identical generalisation properties as when training from small weights (Fig. 3.12G-H).

We further demonstrate that any alteration to the second task's input-output correlation that preserves singular values or singular vectors from the first task leaves those components unchanged during learning (Appendix A.4.5). Based on this observation and drawing inspiration from collaborative work on replay detection in the dorsolateral striatum (not included in this thesis) [275], a new prioritised replay scheme is developed, which contradicts the one proposed by McClelland et al. [191, 192]. In the standard training setup, the time complexity is $\mathcal{O}(N_{\text{modes}}^{\text{total}} \times N_{\text{epochs}})$, where $N_{\text{modes}}^{\text{total}}$ denotes the total number of modes. In contrast, the prioritised replay mechanism reduces this to $\mathcal{O}(N_{\text{modes}}^{\text{modified}} \times N_{\text{epochs}})$, effectively saving $\mathcal{O}(N_{\text{modes}}^{\text{unmodified}} \times N_{\text{epochs}})$ training steps, where $N_{\text{modes}}^{\text{unmodified}} = N_{\text{modes}}^{\text{total}} - N_{\text{modes}}^{\text{modified}}$. This demonstrates that the prioritised replay retains the original learning dynamics while significantly improving efficiency (see Appendix A.4.5).

In summary, having incorporated structured knowledge before revision does not speed up or even slows down learning in comparison to learning from random zero-balanced weights. Notably, the statement holds even when task structures nearly match.(Fig. 3.10B and Fig. 3.12D).

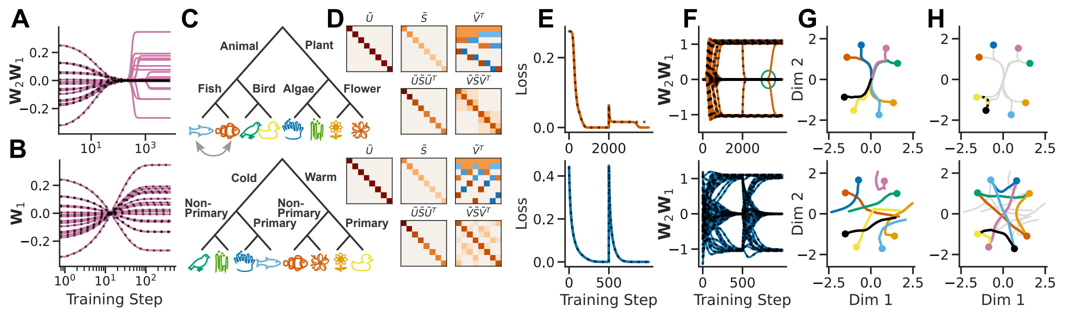


Figure 3.12: Reversal learning and revising structured knowledge. The scale of the x-axis varies in the top and bottom rows. **A** Analytical (black dotted) and numerical (solid) learning dynamics of a reversal learning task. The analytical solution gets stuck on a saddle point, whereas the numerical simulation escapes the saddle point and converges to the target. **B** In a shallow network, training on the same task as in A converges analytically (black dotted) and numerically (solid). **C** Semantic learning tasks. Revised living kingdom (top) and colour hierarchy (bottom). **D** SVD of the input-output coreelation of the tasks and respective RSMs. **E** Analytical (black dotted) and simulation (solid) loss and **F** learning dynamics of first training on the living kingdom (Fig. 3.10A) and subsequently on the respective task in C. The analytical solution fails for the revised animal kingdom as it gets stuck in a saddle point, while the simulation escapes the saddle (top, green circle). Initial training on the living kingdom task from large initial weights and subsequent training on the colour hierarchy have similar convergence times (bottom) **G** Multidimensional scaling (MDS) of the network function for initial training on the living kingdom task from small (top) and large initial weights (bottom). Note how, despite the seemingly chaotic learning dynamics when starting from large initial weights, both simulations learn the same representation. **H** MDS of subsequent training on the respective task in C. For simulation details, see Appendix A.5.14.

3.6 Discussion

We derive exact solutions to the learning dynamics within a tractable model class: deep linear networks. While our findings extend the range of analytically describable two-layer linear network problems, they are still limited by a set of assumptions. In particular, further relaxing the assumptions that input covariance must be whitened and that initialisation must be λ*-balanced* could bring the analysis closer to practical applications in machine learning and neuroscience. Moving towards the nonlinear setting would also make the findings more applicable to real-world scenarios. Despite these limitations, our solutions provide valuable insights into network behaviour. We demonstrate how prior knowledge influences learning dynamics, where notably, this prior knowledge can be interpreted either as the initialisation or as the representation learned from a previous task, as in continual learning. Specifically, we examine the transition between the *rich* and *lazy* regimes by analysing the dynamics as a function of λ—the *relative scale*—across its full range from positive to negative infinity. Our analysis highlights the critical role of the *relative scale*, λ, in governing the transition between *rich* and *lazy* learning regimes. In particular, we uncover a structured *lazy* regime that facilitates transfer learning and a *rapid rich* regime with exponential dynamics. Building on prior work [158], which demonstrated the relevance of these findings in both basic nonlinear models and practical applications, our theoretical framework suggests that exploring unbalanced initialisation strategies may further enhance efficient feature learning. Additionally, we show that this transition is shaped by a nuanced interplay among network architecture, *relative scale* and *absolute scale* -extending the analysis beyond the sole effect of *absolute scale* at initialisation. Finally, we demonstrate the applicability of our solutions to both neuroscience and machine learning, offering exact results for continual learning dynamics, reversal learning, transfer learning, fine-tuning, and the acquisition of structured knowledge. We leave to future work the extension of these initialisation principles to deep networks, as well as the application of our framework to the dynamics of fine-tuning and linear auto-encoders.

Chapter 4

From Lazy to Rich: Beyond linear networks

But linearity is often an approximation to a more complicated reality.

– Steven H. Strogatz

This chapter discusses the work presented in publications [134, 158].

4.1 Introduction

It has been shown that neurons do not exclusively perform linear computations, emphasising the need to explore nonlinear models [46]. In fact, most real-world models naturally incorporate nonlinearities. As such, our objective is to examine the impact of prior knowledge within nonlinear networks. Thus far, we have explored the dynamics of linear networks and examined how initialisation, architecture, and dynamics influence the learning regime and its representation. In this chapter, we move beyond linear networks. We begin by investigating the same parameters (initialisation and architecture) in a two-layer perceptron in a teacher-student setup and its applications. We then analyse how these theoretical results manifest in networks used in practice. Here to investigate the learning regime of the network, we examine the representation it has developed after learning a task.

Theories of representation in biological neural networks range from highly localised representations in single neural units [26] to fully distributed or shared representations [122]. While shared representations offer greater robustness, specialised representations allow for more efficient encoding of information. Experimental evidence supports both ends of this spectrum, with different brain areas and tasks exhibiting distinct forms of representation [35, 221, 100, 128, 20]. Similarly, artificial neural networks display both shared [163, 86, 303] and specialised representations [305, 288], with recent advances in explainable AI, such as the Golden Gate Claude model [274], exemplifying an extreme case of specialisation. Given the trade-off between shared and specialised representations, a critical research challenge is understanding how to guide neural networks toward one form or the other. Previous research [51, 99, 36] has shown that by interpolating between learning regimes, one can transition from shared representations—characterised by random projections in neural tangent kernels—to effective feature learning [272, 158, 297, 67, 285]. In this chapter, we examine the role of initialisation in guiding neural networks toward either specialised or shared representations, offering a complementary perspective on both the lazy learning regime [130] and the rich learning regime [51]. While our analysis is situated within the feature learning regime, it adopts a distinct theoretical approach to characterizing the variety of such regimes by examining how initialization affects representations in standard synthetic neural network framework.

These representations are particularly relevant in contexts such as disentangled representation learning [31] and multi-task learning [45], including continual and transfer learning. Specialised representations can facilitate faster adaptation and reduce catastrophic forgetting [193, 227] by enabling networks to efficiently rewire themselves [267]. Caruana's seminal work on multi-task learning [45] highlighted the advantages of specialisation in improving performance across multiple tasks.

Recent efforts to mitigate catastrophic forgetting [212, 60] have led to the development of regularisation strategies that encourage specialisation, such as elastic weight consolidation (EWC) [148], synaptic intelligence [306], and learning without forgetting [172]. In disentangled representation learning, Locatello et al. [177]

demonstrated that, despite the success of unsupervised approaches, disentanglement does not emerge naturally without an explicit inductive bias, underscoring the need for supervision in enforcing structured representations.

Main Contributions

Our exploration reveals how initialisation biases learning dynamics toward either specialised or shared representations, offering new insights into learning dynamics in over-parameterised networks. Specifically, our work makes the following key contributions:

- We study the impact of initialisation in high-dimensional mean-field neural networks trained with stochastic gradient descent [238, 236, 34].
- Our findings challenge existing assumptions about the relationship between task similarity and catastrophic forgetting [225, 167, 168].
- We identify specific initialisation schemes that promote specialisation by increasing the entropy of readout weights and creating an imbalance between the first and last layers, aligning with findings from Chapter 3 [67].
- We demonstrate the practical implications of our results for regularisation strategies in continual learning, specifically analysing how EWC [148] is influenced by specialisation dynamics and highlighting potential pitfalls in regularisation methods for continual learning.
- Finally, we show how this imbalance impacts neural network dynamics in practical settings.

We begin by introducing the teacher-student setup and the concept of specialisation within this framework while situating our work in the relevant literature. We then explore the application of our findings to the continual learning problem, revisiting existing theoretical frameworks and demonstrating how certain conclusions may not hold under specific initialisation schemes. We conclude this chapter by discussing the implications of our findings for EWC-based mitigation strategies. Finally, we reflect on the limitations of our work and propose future research directions.

4.2 Preliminaries

4.2.1 Notation

In this chapter, we use bold for vectors and matrices, unless we use indices to express the dimensions of the variable. We also

- $\mathbf{W}$ input to hidden weights of student
- $\mathbf{h}^{(1)}$ hidden to output weights (head weights) of student (for teacher 1)
- $\mathbf{h}^{(2)}$ hidden to output weights (head weights) of student (for teacher 2)
- $\mathbf{W}_T^{(1)}$ input to hidden weights of teacher 1
- $\mathbf{W}_T^{(2)}$ input to hidden weights of teacher 2
- $\mathbf{h}_T^{(1)}$ individual hidden to output weights (head weights) of teacher 1
- $\mathbf{h}_T^{(2)}$ individual hidden to output weights (head weights) of teacher 2
- ϕ activation function
- η learning rate of the head weights
- γ controls the correlation between tasks
- N input dimension
- i, j, k denote student indices

4.2.2 The two-layer teacher-student setup for continual learning

The teacher-student framework is a generative model that allows for the controlled creation of synthetic datasets [97]. The framework involves two classifiers: the *teacher* and the *student*, for instance, neural networks as exemplified in Fig. 4.1a. The teacher has fixed randomly drawn weights and maps random inputs $\mathbf{x}$ from a given distribution to labels, providing a rule for generating labels. The student, on the other hand, updates its parameters through learning protocols like stochastic

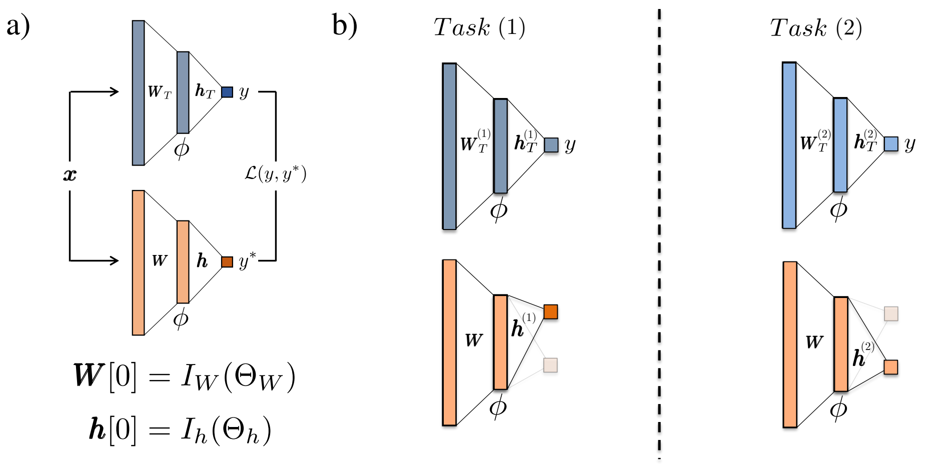


Figure 4.1: Teacher-Student Framework. (a) A standard teacher-student configuration, where a "student" network is trained on i.i.d. inputs using labels provided by a fixed "teacher" network. The student's initial weights, I_W and I_h, are determined by the initialization parameters Θ_W and Θ_h. See Appendix B.3.1 for further details. (b) To model continual learning, we train a two-layer student network sequentially on two different teacher networks, each corresponding to a separate task (Task 1 and Task 2). We refer to this setup as the multi-head configuration, where the student shares the same input pathway but has separate output heads to learn from Teacher 1 and Teacher 2, respectively. See Appendix B.3.1 for details.

gradient descent (SGD) to approximate the teacher's outputs.

We use a teacher-student framework designed for the continual learning setting, shown in Fig. 4.1b, which has been analysed in [167, 168]. This model consists of two randomly initialised teacher networks—one for an upstream task and one for a downstream task.

Each teacher is represented by two-layer neural networks with p^* hidden units and weights $\mathbf{W}_T^{(1)}$, $\mathbf{h}_T^{(1)}$ for the upstream task, and $\mathbf{W}_T^{(2)}$, $\mathbf{h}_T^{(2)}$ for the downstream task. Given a random input $\mathbf{x} \in \mathbb{R}^d$, drawn i.i.d. from a Gaussian distribution $x_i \sim \mathcal{N}(0,1)$, the teachers generate labels (scalar) according to the equation:

$$y^{(t)} = \mathbf{h}_T^{(t)} \cdot \phi\left(\frac{\mathbf{W}_T^{(t)}\mathbf{x}}{\sqrt{d}}\right) \quad \text{for } t = 1,2, \tag{4.1}$$

where ϕ is a non-linear activation function, chosen here as $\phi(z) = \mathrm{erf}\left(z/\sqrt{2}\right)$ to allow the exact analytical evaluation of Gaussian integrals. This setup allows us to generate two datasets $\mathcal{D}^{(1)}$ and $\mathcal{D}^{(2)}$, with controlled similarity between the tasks by manipulating the teacher weights. Specifically, we generate $\mathbf{W}_T^{(1)}$, $\mathbf{h}_T^{(1)}$, and $\mathbf{h}_T^{(2)}$ with i.i.d. Gaussian entries, while $\mathbf{W}_T^{(2)}$ is generated as:

$$\mathbf{W}_T^{(2)} = \gamma \mathbf{W}_T^{(1)} + \sqrt{1-\gamma^2}\mathbf{W}_T^{(\mathrm{aux})}, \tag{4.2}$$

where $\mathbf{W}_T^{(\mathrm{aux})}$ is an auxiliary weight matrix, and γ controls the correlation between tasks. The student is a two-layer neural network with p hidden units, using the same non-linearity ϕ. It is trained using online stochastic gradient descent on a squared error loss, with a shared first-layer weight matrix $\mathbf{W}$ and task-specific readout weights $\mathbf{h}^{(1)}$ and $\mathbf{h}^{(2)}$. Sharing the first-layer weights facilitates the study of continual learning by allowing the model to share features common to multiple tasks, thereby impacting knowledge transfer and forgetting. For both layers, the initial weights are sampled i.i.d. from a Gaussian distribution, with the first-layer weights $\mathbf{W}$ having standard deviation σ_W. This initialisation is generally selected to ensure analytical tractability. The updates for $\mathbf{W}$ and $\mathbf{h}^{(t)}$ at iteration e, under SGD on the squared error loss, are given by:

$$\mathbf{W}[e+1] = \mathbf{W}[e] - \frac{\eta}{\sqrt{d}}\left(\mathbf{h}^{(t)} \cdot \phi\left(\frac{\mathbf{W}\mathbf{x}}{\sqrt{d}}\right) - y^{(t)}\right)\phi'\left(\frac{\mathbf{W}\mathbf{x}}{\sqrt{d}}\right)\mathbf{v}^{(t)}\mathbf{x}, \tag{4.3}$$

$$\mathbf{h}^{(t)}[e+1] = \mathbf{h}^{(t)}[e] - \frac{\eta}{d}\left(\mathbf{h}^{(t)} \cdot \phi\left(\frac{\mathbf{W}\mathbf{x}}{\sqrt{d}}\right) - y^{(t)}\right)\phi\left(\frac{\mathbf{W}\mathbf{x}}{\sqrt{d}}\right), \tag{4.4}$$

where η is the learning rate and $y^{(t)}$ is the target output from the teacher network for task t. In the large input dimension limit $d \to \infty$ while keeping the other dimensions finite, we can derive closed-form expressions for the evolution of the order parameters

$$\mathbf{Q} = \frac{1}{d}\mathbf{W}\mathbf{W}^{\intercal}, \quad \mathbf{R}^{(t)} = \frac{1}{d}\mathbf{W}\mathbf{W}_T^{(t)\intercal}, \quad \mathbf{T}^{(t,t')} = \frac{1}{d}\mathbf{W}_T^{(t)}\mathbf{W}_T^{(t')\intercal}, \quad \mathbf{h}^{(t)}, \quad \mathbf{h}_T^{(t)}; \tag{4.5}$$

where $t, t' \in \{1,2\}$ refer to the two tasks. Intuitively, $\mathbf{Q}$ measures the *self-overlap* or internal structure of the current weights $\mathbf{W}$. It captures how the learned features (rows of $\mathbf{W}$) are organised, representing the network's internal representation. $\mathbf{R}^{(t)}$ quantifies the *alignment between the current weights* $\mathbf{W}$ and the *target weights* $\mathbf{W}_T^{(t)}$ for task t. A higher value indicates that the network is better aligned with task t. $\mathbf{T}^{(t,t')}$ quantifies the *similarity between the first-layer weights* of teachers t and t'. $\mathbf{h}^{(t)}$ and $\mathbf{h}_T^{(t)}$ correspond to the student and the teacher head weights for task t, respectively. Together, $\mathbf{T}^{(t,t')}$ and the teacher head weights $\mathbf{h}_T^{(t)}$ and $\mathbf{h}_T^{(t')}$ provide a measure of how similar the two tasks t and t' are.

We use the parameter update expressions from Equations 4.3 to derive the corresponding updates for the order parameters. In the high-dimensional limit ($d \to \infty$), these discrete update equations converge to ordinary differential equations (ODEs), which can be integrated either numerically or analytically in certain cases [132]. As is often the case in the statistical physics of disordered systems, this approach was first derived non-rigorously by Saad and Solla [238] and Biehl and Schwarze [34], with later works laying down a mathematical foundation showing the concentration of the ODEs [107, 29]. Let us define the pre-activations of the student and task-t teacher given an input $\mathbf{x}$ from task t as

$$\lambda_i = \frac{1}{\sqrt{d}}\mathbf{W}_i \cdot \mathbf{x}, \qquad \rho_i^{(t)} = \frac{1}{\sqrt{d}}\mathbf{W}_{T,i}^{(t)} \cdot \mathbf{x}, \tag{4.6}$$

and denote the difference between the teacher and student predictions by $\Delta^{(t)} = \mathbf{h}^{(t)} \cdot \phi(\lambda) - \mathbf{h}_T^{(t)} \cdot \phi(\rho)$. The corresponding ODEs for the order parameters in the limit $d \to \infty$ are given by:

$$\begin{aligned}
\frac{dQ_{ik}}{d\tau} &= -\eta h_i^{(t)} \langle \phi'(\lambda_i)\Delta^{(t)}\lambda_k\rangle - \eta h_k^{(t)} \langle \phi'(\lambda_k)\Delta^{(t)}\lambda_i\rangle \\
&+ \eta^2 h_i^{(t)} h_k^{(t)} \langle \phi'(\lambda_i)\phi'(\lambda_k)(\Delta^{(t)})^2\rangle, && (4.7)\\
\frac{dR_{in}^{(t')}}{d\tau} &= -\eta h_i^{(t)} \langle \phi'(\lambda_i)\Delta^{(t)}\rho_n^{(t')}\rangle, && (4.8)\\
\frac{dh_i^{(t)}}{d\tau} &= -\eta \langle \Delta^{(t)}\phi(\lambda_i)\rangle, && (4.9)
\end{aligned}$$

where $\tau = \text{epoch}/d$ represents continuous time in the high-dimensional limit, and

$t, t' \in 1, 2$ denote the task indices. The angular brackets indicate an average over the pre-activations. The pre-activations themselves are centred Gaussian random variables with covariances determined by the order parameters $\mathbf{Q}$, $\boldsymbol{R}^{(t)}$, and $\mathbf{T}$ such that

$$P(\beta,\gamma) = \frac{1}{(2\pi)^{F+H}|\tilde{C}|^{1/2}} \exp\left(-\frac{1}{2}(\beta,\gamma)^T\tilde{C}^{-1}(\beta,\gamma)\right), \tag{4.10}$$

$$C = \begin{pmatrix} Q & R^{(1)} & T^{(2,1)} \\ R^{(1),T} & T^{(1,1)} & R^{(2)} \\ T^{(1,2),T} & R^{(2),T} & T^{(2,2),T} \end{pmatrix}. \tag{4.11}$$

These averages can be computed analytically for certain activation functions. For instance, in the case of a rescaled error function introduced in the main text [238, 34], the relevant averages are given by:

$$I_2 = \langle\phi(\beta)\phi(\gamma)\rangle = \frac{1}{\pi}\arcsin\left(\frac{\Sigma_{12}}{\sqrt{(1+\Sigma_{11})(1+\Sigma_{22})}}\right), \tag{4.12}$$

$$I_3 = \langle\phi'(\zeta)\beta\phi(\gamma)\rangle = \frac{2\Sigma_{23}(1+\Sigma_{11}) - 2\Sigma_{12}\Sigma_{13}}{\sqrt{\Lambda_3}(1+\Sigma_{11})}, \tag{4.13}$$

$$I_4 = \langle\phi'(\zeta)\phi'(\iota)\phi(\beta)\phi(\gamma)\rangle = \frac{4}{\pi^2\sqrt{\Lambda_4}}\arcsin\left(\frac{\Lambda_0}{\sqrt{\Lambda_1\Lambda_2}}\right), \tag{4.14}$$

where the Greek letters represent arbitrary pre-activations with covariance matrix $\mathbf{\Sigma}$, and the auxiliary quantities Λ_i are given by:

$$\Lambda_0 = \Lambda_4\Sigma_{34} - \Sigma_{23}\Sigma_{24}(1+\Sigma_{11}) - \Sigma_{13}\Sigma_{14}(1+\Sigma_{22}) + \Sigma_{12}\Sigma_{13}\Sigma_{24} + \Sigma_{12}\Sigma_{14}\Sigma_{23}, \tag{4.15}$$

$$\Lambda_1 = \Lambda_4(1+\Sigma_{33}) - \Sigma_{23}^2(1+\Sigma_{11}) - \Sigma_{13}^2(1+\Sigma_{22}) + 2\Sigma_{12}\Sigma_{13}\Sigma_{23}, \tag{4.16}$$

$$\Lambda_2 = \Lambda_4(1+\Sigma_{44}) - \Sigma_{24}^2(1+\Sigma_{11}) - \Sigma_{14}^2(1+\Sigma_{22}) + 2\Sigma_{12}\Sigma_{14}\Sigma_{24}, \tag{4.17}$$

$$\Lambda_3 = (1+\Sigma_{11})(1+\Sigma_{33}) - \Sigma_{13}^2. \tag{4.18}$$

$$\Lambda_4 = (1+\Sigma_{11})(1+\Sigma_{22}) - \Sigma_{12}^2. \tag{4.19}$$

The ODEs governing the dynamics of the order parameters in the continual learning setting were first derived by Lee et al. [167]. They provide a comprehensive analytical framework for tracking the student network's learning dynamics and the emergence of specialisation throughout training. For example, we can formulate the generalisation error in terms of the order parameters and track its dynamics through time. Let us begin by

$$\varepsilon^{(t)} = \frac{1}{2}\mathbb{E}_{\mathbf{x}}\left[\left(\mathbf{h}^{(t)} \cdot \phi\left(\frac{\mathbf{W}\mathbf{x}}{\sqrt{d}}\right) - y^{(t)}\right)^2\right] \tag{4.20}$$

$$\begin{aligned}
&\varepsilon^{(t)} = \\
&\frac{1}{2}\left[\sum_{i,k} h_i^{(t)} h_k^{(t)} \phi(\lambda_i)\phi(\lambda_k) + \sum_{m,n} h_{T,m}^{(t)} h_{T,n}^{(t)} \phi(\rho_m)\phi(\rho_n) - 2\sum_{i,n} h_i^{(t)} h_{T,n}^{(t)} \phi(\lambda_i)\phi(\rho_n)\right]. \\
&= \frac{1}{2}\sum_{i,k} h_i^{(t)} h_k^{(t)} I_2(i,k) + \frac{1}{2}\sum_{n,m} h_{T,n}^{(t)} h_{T,m}^{(t)} I_2(n,m) - \sum_{i,n} h_i^{(}t) h_{T,n}^{(}t) I_2(i,n) \\
&= I_{21}(\mathbf{Q}, \mathbf{h}^{(t)}) + I_{21}(\mathbf{T}^{(t,t)}, \mathbf{h}_T^{(t)}) - \frac{1}{2} I_{22}(\mathbf{Q}, \mathbf{R}^{(t)}, \mathbf{T}^{(t,t)}, \mathbf{h}^{(t)}, , \mathbf{h}_T^{(t)}),
\end{aligned} \tag{4.21}$$

where we have expressed the equation in terms of I_{21} and I_{22}, making their dependence on the previously defined order parameters explicit.

In the context of continual learning, we define *forgetting* on the first task as the change in generalisation error following a task switch at time $\tilde{s}$. Specifically, we quantify forgetting at time $\tilde{s} + s$ as:

$$\text{Forgetting:} \quad F_t \equiv \varepsilon^{(t)}_{|\tilde{s}+s} - \varepsilon^{(t)}_{|\tilde{s}} \tag{4.22}$$

Here, $\varepsilon^{(t)}$ denotes the generalization error on task t. For task 1, an increase in this error after the switch indicates *positive forgetting*, reflecting a degradation in performance due to interference from the newly learned task.

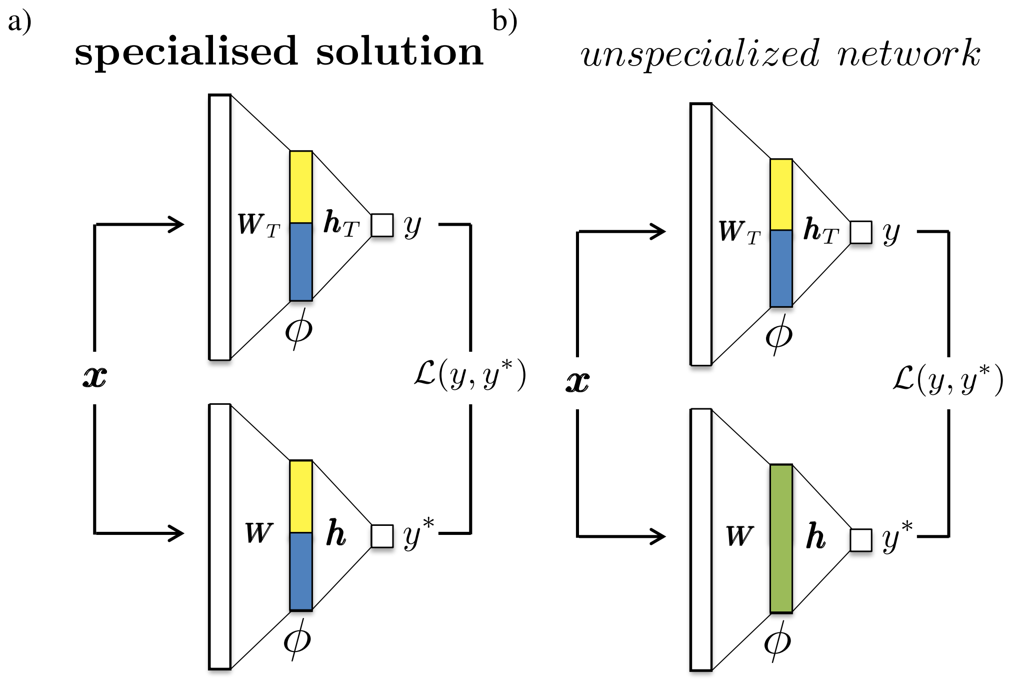


Figure 4.2: Specialisation. Schematic illustration of specialisation in the student-teacher setup. Specialisation is measured by comparing the similarity of hidden layer representations between the student and teacher. (a) In the specialised setting, each student neuron (yellow and blue) aligns with a specific teacher neuron (yellow and blue). (b) In the non-specialised scenario, the student redundantly shares teacher representations across multiple neurons (green). See Appendix B.3.2 for details.

4.3 Rich and Lazy learning

We start by exploring the representations developed during the initial phase of learning in the continual learning setting. Our analysis remains within the feature learning regime and provides a clearer definition of the different rich representations that emerge as a function of initialisation, with a particular focus on specialisation. This setup lends itself to a theoretical approach that examines how initialisation within standard synthetic neural network frameworks influences specialisation.

4.3.1 Specialisation: Student-teacher setting

While a detailed quantitative characterisation of specialisation follows in the next section, we briefly introduce the concept within the teacher-student framework. Saad et al. [238] demonstrated that when both the teacher and student are represented as committee machines—where the first layer's weights remain fixed and learning

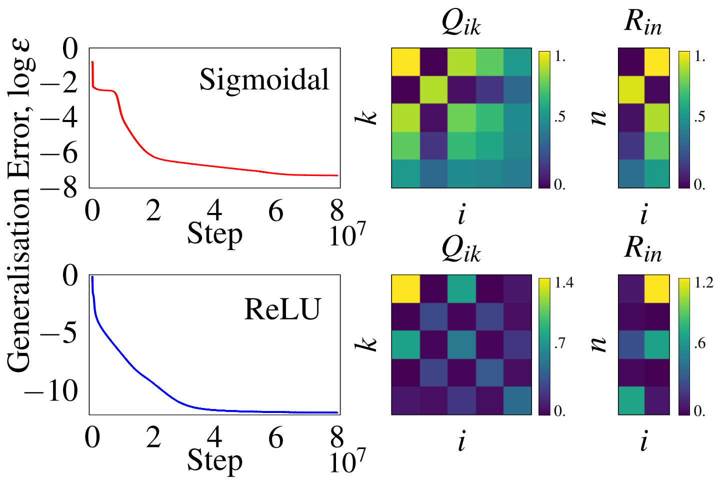


Figure 4.3: Initialisation impacts specialisation. Previous work established a relationship between the activation function ϕ and the propensity for the student nodes to specialise to teacher nodes. However we show in this work that this is an overly simplistic description; other factors including student weight initialisations I_W, I_h, parameterised by Θ_W, Θ_h arguably play a stronger role. Generalisation error curves for two simulations of the teacher-student setup, one with a ReLU activation function and one with a scaled error activation function. Θ_W and Θ_h are chosen to achieve a solution with ReLU that specialises—as indicated by sparser overlap matrices on the bottom right, and a scaled error function solution that does not specialise—as indicated by denser overlap matrices on the top right. A sparse (dense) Q matrix shows few (many) nodes are active, while a sparse (dense) R matrix shows student nodes are representing teacher nodes in a targeted (redundant) manner. Further details for the quantities described can be found in subsection 4.4.1. The teacher network comprises two hidden units with weights initialised from a Gaussian distribution, while the student network contains four hidden units. Further simulation details are provided in Appendix B.3.3.

occurs in the second layer—each student neuron becomes specialised by aligning with a particular teacher neuron. Similarly, Goldt et al. [106] observed that for sigmoidal activation functions in two-layer networks, an over-parametrised student will selectively use only a subset of those units to replicate the teacher's outputs. This phenomenon, termed specialisation, stands in contrast to a student redundantly sharing representations of the teacher across neurons as depicted in Fig. 4.2.

In this section, we present a more comprehensive account of the factors underlying specialisation. In contrast to Goldt et al. [106], we argue that initialisation -not the activation function -is chiefly responsible. We highlight this in Fig. 4.3, by showing that with carefully chosen initialisations we can train a highly specialised ReLU student (bottom panels), and a non-specialising sigmoidal student (top panels)–shown by the sparser Q and R matrices of the ReLU network–which represents the opposite of the conclusions presented in Goldt et al. [106]. We begin by aiming to establish what properties of an initialisation promote specialisation.

4.3.1.1 Specialisation's relevance for continual learning

In this section, we formalise the relationship between specialisation and initialisation. A student can effectively ignore a unit in two ways: either the unit's post-activation is near 0 (inactive) or the corresponding second-layer weight is 0. This motivates three measures for specialisation based on the definition of entropy- over the hidden units, head weights, and the product of both:

$$H_h = -\sum_i^p |\tilde{h}_i| \log |\tilde{h}_i|, \quad H_Q = -\sum_i^p \tilde{Q}_{ii} \log \tilde{Q}_{ii}, \quad H_m = -\sum_i^p \tilde{Q}_{ii} |\tilde{h}_i| \log(\tilde{Q}_{ii} |\tilde{h}_i|); \tag{4.23}$$

where the tilde denote normalisation, i.e. $|\tilde{h}_i| = \frac{|h_i|}{\sum_i^P |h_i|}$, $\tilde{Q}_{ii} = \frac{Q_{ii}}{\sum_i^P Q_{ii}}$ and p is the number of hidden neuron. Each of the quantities H_h, H_Q, and H_m can be interpreted as a form of Shannon entropy, which characterises the uncertainty or spread within a probability distribution.

The term H_h measures the entropy of the normalised magnitudes $|\tilde{h}_i|$. A high value of H_h indicates that the $|h_i|$ values are distributed relatively evenly across units, corresponding to a state of low specialisation. Conversely, a low value of H_h reflects a situation where one or a few $|h_i|$ dominate, indicating strong specialisation.

Similarly, H_Q quantifies the entropy of the normalised diagonal elements of the student overlap matrix Q. In this context, Q is the order parameter that quantifies the overlap of the student's hidden representations used in statistical physics to analyse the dynamics of generalisation error. Here, we focus only on the diagonal elements of Q, Q_{ii} that reflect the mean squared magnitude of the input weights for the i-th

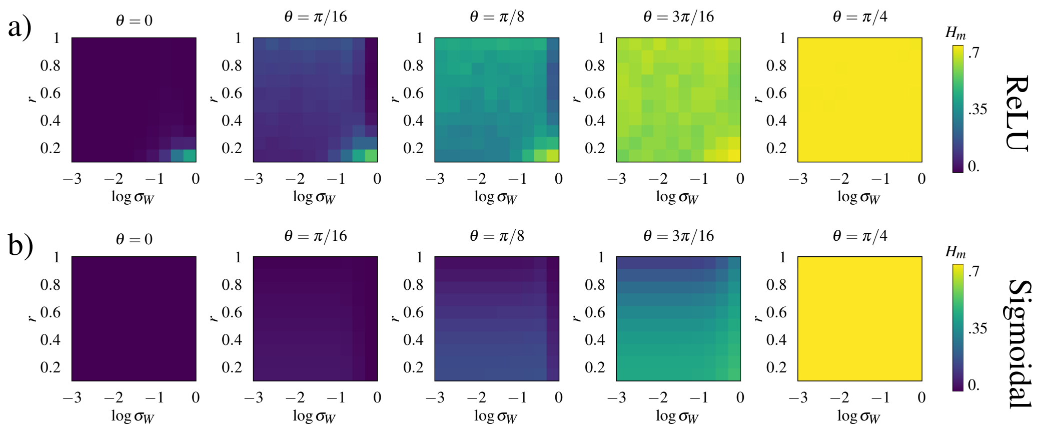


Figure 4.4: Phase diagrams show significance of initialisation for specialisation. The phase diagrams show with colour the aggregated entropy Eq. 4.23 evaluated for different initialisations. On the x-axis we span over the standard deviation of the first layer. The second layer is initialised using polar coordinates, and the y-axis represents the norm while the different panels give the angle spanning from orthogonal units ($\theta = 0$) to identical units ($\theta = \pi/4$). Specialisation is achieved by blue-leaning initialisations, while yellow-leaning ones exhibit high entropy and therefore non-specialised solutions. The teacher has one hidden unit, while the student has two hidden units. For simulation details, see Appendix B.3.4.

neuron and serve as a proxy for measuring entropy. A high value of H_Q indicates that the overlap is evenly distributed across units, implying low specialisation, whereas a low value of H_Q suggests that the overlap is concentrated in a subset of units.

The quantity H_m represents a mixed entropy that combines information from both Q and h. Specifically, it measures the entropy of the product $\tilde{Q}_{ii}|\tilde{h}_i|$ across units. This measure captures joint specialisation: H_m is minimised when both the overlap Q and the activations h are highly concentrated together, and maximised when both are broadly distributed.

All together, maximum entropy in these measures corresponds to no specialisation, while minimum entropy corresponds to maximum specialisation.

We can investigate how these measures vary as a function of different properties of the problem setup, in particular those related to initialisation. For simplicity, we start by considering the case where the teacher network has a single hidden neuron while the student network contains two hidden units. This allows us to initialise the second layer weights of the student in polar coordinates, with precise and interpretable control over the scale and asymmetry of weights. Formally, we parameterise our readout initialisations according to

$$\mathbf{h}^{(t)}[0; r^{(t)}, \theta^{(t)}] = (r^{(t)} \cos\theta^{(t)}, r^{(t)} \sin\theta^{(t)}). \tag{4.24}$$

Fig. 4.4 contains phase diagrams showing how the entropy measures in Eqs. 4.23 vary with the initialisation parameters $r^{(t)}$, $\theta^{(t)}$, and σ_W. We can make several observations: (i) the strongest determinant of specialisation is the asymmetry in the second layer weights, i.e. the θ parameter. (ii) This is the case for both ReLU and sigmoidal activation functions, reinforcing the point made in the example from Fig. 4.3. (iii) the scale of initialisations (parameters σ_W, r) are also important. The results show that a positive norm imbalance (that is, $\sigma_W < r$) fosters greater specialisation in non-linear networks. Notably, these findings contrast with those for linear networks, where a positive λ results in lazy learning—an outcome not expected to exhibit any specialisation, as detailed in Chapter 3.

Overall, while most prior work adopts a similar approach to initialising the readout weights, we propose a novel scheme based on polar coordinates. This formulation enables a smooth transition between specialised and distributed representations. It is essential to acknowledge the limitations of the current setup, which has been primarily demonstrated with just two hidden neurons. As the number of neurons increases, maintaining entropy control and achieving balance becomes increasingly challenging, making it harder to draw definitive conclusions. In the following section, we explore how our findings might generalise to more complex architectures (e.g. more hidden neurons). However, as previously noted, the distinction between rich and lazy regimes may lose clarity or consistency in these higher-dimensional settings. In particular, we identify initialisation strategies that promote specialisation by increasing the entropy of the readout weights and introducing an asymmetry between the first and last layers. These findings further underscore the importance of layer-wise imbalance in shaping the learning regime, echoing similar insights from Chapter 3 [67]. This further highlights that layer imbalance plays a crucial role in shaping both the learning regime and the representations developed in simple non-linear networks, as seen through the lens of specialisation.

4.4 Applications

In this section, we explore the practical implications of our findings on the relationship between initialisation and specialisation, particularly in the contexts of continual learning. We further illustrate how the resulting imbalances affect neural network behavior in practical scenarios. All together, we show that the relative scale (imbalance) also impact the degree of specialisation and feature learning in practice.

4.4.1 Continual learning

A fundamental challenge in continual learning is the mitigation of catastrophic forgetting, wherein performance on previously acquired tasks deteriorates upon learning new ones [193, 227]. While early work, such as that by Goodfellow et al. [110], typically assumed a monotonic relationship between task similarity and forgetting, more recent studies have revealed a richer and more nuanced set of forgetting profiles. In particular, Ramasesh et al. [225] and Lee et al. [167] conducted systematic experiments across diverse architectures and training paradigms, showing that catastrophic forgetting is most pronounced between tasks of intermediate similarity. To account for this phenomenon, Lee et al. [168] proposed a mechanistic explanation grounded in neural resource allocation. They posited a trade-off between the reuse of previously specialized neurons and the recruitment of dormant units. In particular, task-specific specialization often leaves unused capacity in the form of dormant units, which can be recruited when tasks are highly dissimilar—thereby helping the network preserve performance across multiple domains [45]. When task similarity is intermediate, however, the network tends to partially reuse existing neurons, increasing the risk of interference and forgetting. This dynamic reflects a fundamental tension between efficiency and flexibility. This phenomenon is evocatively captured by Maslow's cognitive bias: "If the only tool you have is a hammer, it is tempting to treat everything as if it were a nail" [211]. Over-reliance on existing representational "tools" (neurons) can induce interference, whereas under-utilization of them may result in inefficient or unstable representations. We refer to this characteristic pattern—where forgetting peaks at intermediate task similarity—as Maslow's Hammer

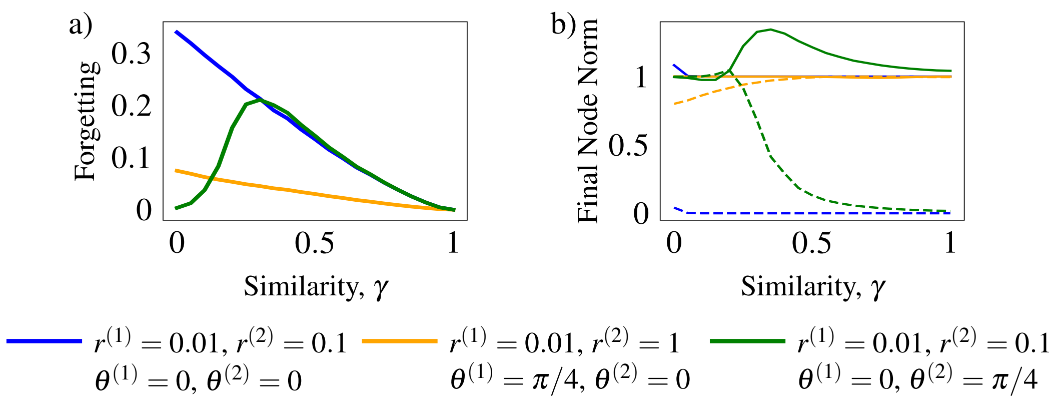


Figure 4.5: Initialisation and specialisation properties can influence profile of forgetting vs. similarity. *(a)* forgetting (defined in 4.22) as a function of task similarity (defined in 4.2) can be both monotonic, shown here for the cases of specialisation after both tasks (blue), and no-specialisation + large, asymmetric second head initialisation (orange); *or* non-monotonic (green, as characterised by Maslow's hammer [168]). *(b)* the final norm of the two nodes (one solid and one dashed) is defined as the absolute norm of the weights at the end of training on both tasks as a function of task similarity. In the cases that lead to monotonic forgetting, nodes are fully re-used, either because the corresponding new head is initialised large (orange) or because the new head is symmetrically initialised, and the nodes continue to represent redundant information during the second task (blue). The networks use sigmoidal activations. The teacher has one hidden unit with weights initialised from a zero-mean Gaussian, while the student has two hidden units. For simulation details, see Appendix B.3.5.

profile. Formalizing such non-monotonic forgetting trajectories may be critical for developing more robust and adaptive continual learning systems.

4.4.1.1 Specialisation underlies Maslow's hammer

We aim to revisit the two established forgetting profiles empirically observed in the continual learning literature: namely, the Maslow's Hammer profile, observed empirically first in Ramasesh et al. [225], and the monotonic forgetting profile, more typically assumed and observed in Goodfellow et al. [110]. The phase diagrams in Fig. 4.4 demonstrate that initialisation can drastically change the type of solutions found by the student after training on one teacher. While this may be inconsequential as the generalisation error remains unaffected, in many cases, the specific structure of the learned representation can have a substantial impact on downstream tasks and the extent of forgetting the first task.

In the worst case scenario, the student undergoes no specialisation during the first task. During the second task, there is no notion of the trade-off between node reuse and node activation discussed in [168]; rather, the student continues to find a non-specialised solution to the second teacher, effectively fully reusing its entire representation for the second task. Consequently, the amount of forgetting with respect to the initial task decreases monotonically with task similarity. This extreme case is illustrated in orange in Fig. 4.5. Further, *even with* specialisation after the first task, large asymmetric initialisation in the second task readout weights can induce this monotonic relationship, again by pushing the student into reuse rather than activation (blue Fig. 4.5). Finally, after the network learns a specialised representation during the first task and is initialised symmetrically for the second, we observe the characteristic U-shaped pattern associated with Maslow's hammer—a phenomenon reported in various continual learning settings [225]. This case is illustrated in green in Fig. 4.5.

In a broader context, a rich diversity of behaviours can emerge, driven by factors such as the initialisation schemes, the scale of weights in the first layer, and the readout heads for both tasks. A glimpse of this behavioural diversity is provided in Appendix B.2, where we further explore the interaction between these factors and their impact on forgetting in continual learning. Altogether, these results illustrate the complex relationship between the forgetting profile, initialisation, and specialisation beyond the previously established forgetting profiles.

4.4.1.2 Specialisation underlies EWC

The findings relating specialisation to forgetting from Sec. 4.4.1.1 have direct consequences for interference mitigation strategies such as EWC. EWC is a regularisation-based method that computes a measure of "*importance*" for each weight with respect to a task via the Fisher information [148]. The Fisher information is defined as the expected outer product of the gradient of the log-likelihood, that is,

$$F_{ij} = \mathbb{E}_{x \sim \mathcal{D}} \left[\frac{\partial \log p(y \mid x, \theta)}{\partial \theta_i} \frac{\partial \log p(y \mid x, \theta)}{\partial \theta_j} \right],$$

where θ denotes the model (here the student) parameters, x the input to the model, y the output of the model and the expectation is taken over the data distribution $\mathcal{D}$ corresponding to the task at hand (here the teacher). EWC typically approximates F as a diagonal matrix (that we note F_i), treating each parameter i independently. In practice, we approximate the Fisher information matrix by computing the squared gradients of the student model's loss—evaluated against the teacher's output—and averaging them over the dataset.

Subsequently, a squared penalty scaled by this importance is applied to the deviation of this weight during learning of future tasks as follows:

$$\mathcal{L}_{\text{EWC}}(\mathbf{W}) = \mathcal{L}(\mathbf{W}) + \frac{\xi}{2} \sum_i F_i (W_i - W_i^*), \tag{4.25}$$

where $\mathbf{F}$ is the Fisher information matrix, ξ is a regularisation strength parameter, and $\mathbf{W}^*$ are the weights at the end of training on the first task. The diagonal entries F_i reflect the sensitivity of the model's predictions to perturbations in individual parameters, and are used to selectively constrain important parameters during subsequent learning to mitigate forgetting.

In cases where the network does not specialise, i.e. multiple student nodes learn redundant representations for a given teacher node, the nodes have equal importance. Consequently, EWC cannot distinguish between these sets of weights and depending on the regularisation parameter ξ either lets these nodes move during training on the second task (under-regularises) leading to forgetting, or lets none move (over-regularises) leading to no transfer. We show results illustrating this behaviour in the teacher-student setup in Fig. 4.6. In particular, we show the regime of intermediate task similarity, where in [168] previously argued that EWC should

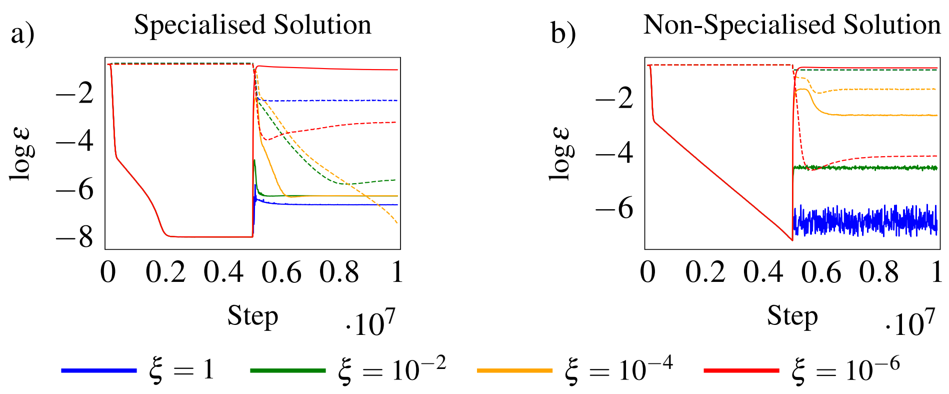


Figure 4.6: EWC is strongly reliant on specialisation. We show the generalisation error in the first (solid line) and second (dashed) task for different EWC regularisation strengths. *(a)* When the student finds a specialised solution to the first task, there is a range of EWC regularisation strength ξ for which the activated units can remain fixed and spare capacity can be used to learn the second task—leading to low generalisation error in both tasks ($\xi = 10^{-2}$, $\xi = 10^{-4}$ perform very well). *(b)* When the student does not specialise in the first task, EWC reduces to an inflexible regulariser that either penalises plasticity everywhere—leading to little forgetting but no further learning (e.g. $\xi = 1$), or does not penalise any plasticity—leading to catastrophic forgetting (e.g. $\xi = 10^{-6}$). The teacher network has a single hidden unit, with weights initialised from a Gaussian distribution with zero mean and unit variance. The student network consists of two hidden units. The input weights are initialised from a Gaussian with mean 0 and standard deviation 0.001. The head of the student network is initialised using the polar method ensuring specialised and non-specialised readouts. For simulation details, see Appendix B.3.6.

perform better than methods such as replay. Altogether, we analysed how EWC [148] is influenced by specialisation dynamics and highlight pitfalls in regularisation methods for continual learning.

4.4.2 Imbalanced initialisations in practice

Until now, our focus has been on simplified networks, gradually increasing their complexity through architectural decisions, initialisation strategies, and the incorporation of non-linearities. We now turn to exploring how our theoretical findings—on specialisation, the rich and lazy regimes, and initialisation—manifest in practical scenarios. We start by investigating how imbalanced initialisation can promote disentangled representations and enhance interpretability in sparse auto encoders, which aligns with the specialisation effects discussed in Sec. 4.3. Next, we consider the implications for grokking and analyse the interpretability of the early layers in a convolutional neural network. Finally, we show that the continual learning behaviours observed in simplified setups also appear in more realistic contexts, such as a continual learning task using MNIST.

4.4.2.1 Increasing specialisation in disentangled representation learning

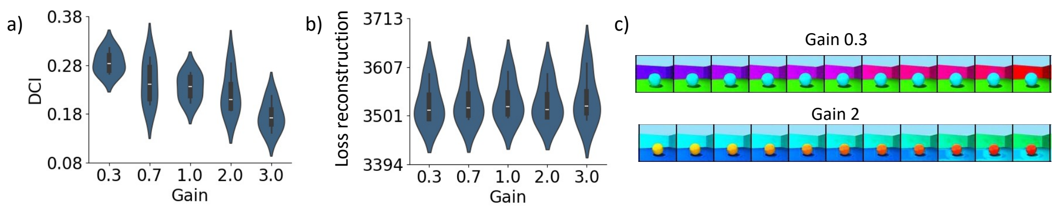


Figure 4.7: Violin plots of *a)* the Disentanglement, Completeness, and Informativenes (DCI) [80] score and *b)* the reconstruction loss against gain of a beta-VAE trained on the 3DShapes data set. The disentanglement score decreases as the gain increases while the reconstruction loss remains steady, *c)* Example traversals of models with gains 2 and 0.3, respectively, highlighting a disentangled dimension for gain 0.3 and a mixed dimension for gain 2. Experimental details can be found in the Appendix B.3.7.

We extend the results on imbalanced initialisation and apply them beyond the limited setting of our framework in the context of disentangled representation learning, where the goal is to separate latent factors. A seminal contribution to this domain came with the β-VAE model, where Higgins et al. [120] demonstrated how increasing the KL-divergence term can enforce disentanglement by encouraging specialised latent representations. Many studies have built upon these foundational frameworks to enhance disentanglement performance, exploring different training regimes [178, 94] and loss functions [49, 145, 154]. Here, we contribute to this literature by applying our theoretical insights and examining the impact of initialisation on disentanglement performance. Specifically, we examine how initialisation impacts specialisation in disentanglement learning on the 3DShapes dataset [41] using the β-VAE model- widely adopted for such tasks [120, 42].

We implement a β-VAE model, employing the ”DeepGaussianLinear” architecture for the decoder and the ”DeepLinear” architecture for the encoder, as specified in [177]. Both architectures are composed of five fully connected layers with ReLU activations. The model is trained using the Adam Optimiser, which optimises a loss function that combines KL divergence and binary cross-entropy-based reconstruction loss. Additional details are given in Appendix B.3.7.

In these experiments, we adjust the variance of the weights in a deep fully-connected encoder, by varying the constant gain of the Xavier initialisation [105]. Specifically, the first block of layers was initialised with gain g, while the readout layer received a gain $1/g$. Notice that $g = 1$ represents the standard initialisation scheme.

Results are shown in Fig. 4.7, despite very similar levels of reconstruction loss, networks initialised with smaller gains improved disentanglement in the β-VAE network, as reflected in higher Disentanglement, Completeness, and Informativeness (DCI) scores [80]. Note that a smaller value of g indicates greater variance in the output layer relative to the input layer, reflecting a positive network imbalance. Therefore, this aligns with the theoretical results presented above and further confirms that modulating the initialisation gain can increase the network's disentanglement. Although the scope of these experiments is limited, they provide preliminary validation of our theoretical framework in more realistic contexts, encouraging further investigation into alternative initialisation schemes with varying levels of imbalance.

We advocate for further investigation into alternative initialisation schemes with varying levels of imbalance. Moreover, we highlight the need for future research to extend these experiments by considering a wider variety of datasets (Car3D [76], dSprites [190],), network architectures (Conv, Linear), initialisation strategies (Gaussian Xavier Initalisation) and different metric (SAP [154, 120],) to fully explore the implications of our findings.

4.4.2.2 Increasing Interpretability using Sparse Auto Encoder

To support our finding that larger hidden-to-output weights promote specialised representation, we examine activation sparsity in a sparse autoencoder, where sparsity reflects specialisation. We predict that increasing the relative weight of later layers will lead to greater sparsity in the autoencoder's activations.

We conduct the following experiment in two phases: *Phase 1:* We train a standard VAE (similar to Sec. 4.4.2.1) on MNIST which was initialised with small weights to ensure the feature learning regime [99] (we sample from a Gaussian with standard deviation 0.001). Importantly, the latent dimension of this VAE is smaller than the input and forms an entangled latent space. *Phase 2:* In a similar manner to the recent approach on the Claude line of Large Language Models [274], we train a sparse autoencoder (SAE) from the latent space of the VAE, with the aim of improving the sparsity and disentanglement of the latent space. For our model, we train in exactly the same manner, except we *do not use the L1 regularisation on the hidden activity*. Thus, for our model, there is no explicit pressure on the autoencoder to embed representations sparsely. For simplicity, we will refer to this model as an implicit Sparse Autoencoder (iSAE). We repeat this process with varying degrees of initialisation imbalance and track the sparsity of the SAE and iSAE. Denoting the hidden layer activity of the networks for the entire MNIST dataset as H, we define an indicator function in Eq. 4.26 for a single neuron responding to a single data point:

$$\mathbf{1}(H_{ij}) =: \begin{cases} 1, & H_{ij} > 0 \\ 0, & \text{otherwise} \end{cases} \tag{4.26}$$

We calculate the sparsity across the dataset as the average number of data points the hidden neurons respond to. The network is initialised with varying values of υ, where increasing υ results in progressively larger initial weights for the decoder relative to the encoder.

The results of this experiment are shown in Fig. 4.8. We see clearly from these

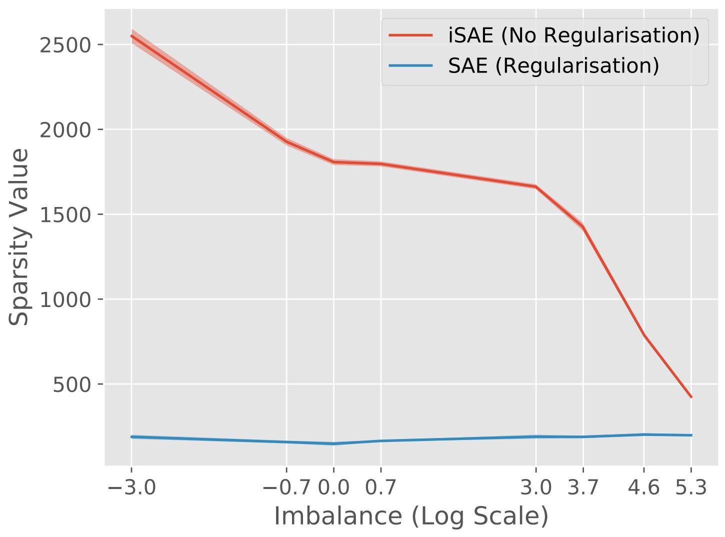


Figure 4.8: Implicit regularisation from initialisation imbalance in a sparse autoencoder:
We track the sparsity of the iSAE and SAE for varying degrees of initialisation imbalance (x-axis). The imbalance on the x-axis depicts the natural log of the imbalance parameter (υ). Thus, 0.0 depicts balanced initialisation typically used in practice. The y-axis depicts the corresponding sparsity calculated using Eq. B.1 in the Appendix. Clearly, as the imbalance increases, the sparsity of the iSAE decreases, while the SAE does not respond due to its explicit regularisation. Results depict the average over ten runs with two standard deviations on either side of the mean. For simulation details, see Appendix B.3.8.

results that as the initialisation imbalance is pushed towards the hidden-to-output weights such that they are larger than the input-to-hidden weights, then the sparsity of the iSAE latent space improves dramatically. This is consistent with the positive imbalance described in Sec. 4.3.1.1, demonstrating alignment between our empirical observations and the minimal model analysis presented above. In contrast, the SAE with explicit sparsity regularisation remains unaffected by varying levels of initialisation imbalance. These results provide empirical support for our predictions and underscore the role of disentanglement in improving both interpretability and generalisation.

4.4.2.3 Interpretability of early layers in CNN

We now shift our focus to a different class of models—convolutional neural networks (CNNs)—to explore how layer-wise imbalances influence representation learning.

As shown in Fig. 4.9, panel (a), filters in CNNs trained on image classification tasks frequently align with edge detectors, consistent with earlier findings [152]. We demonstrate that modifying the learning rate of the first layer—which effectively alters the imbalance between layers—can either strengthen or weaken this alignment. Our results indicate that a positive imbalance (i.e., a positive λ) encourages more robust feature learning in nonlinear networks. This observation is consistent with the specialisation dynamics discussed in Sec. 4.3, and is further formalised in our analysis of two-layer ReLU networks in [158].

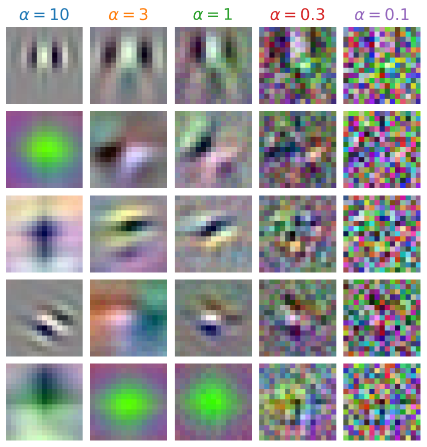
4.4.2.4 Grokking

We now explore the impact of imbalances on grokking. Grokking refers to the phenomenon where networks trained on simple modular arithmetic tasks only generalise after initially memorising their training data [218]. This behaviour is believed to result from a transition between lazy and rich learning [155, 182, 234] and is considered crucial for understanding emergent phenomena [204]. We show in panel b) of Fig. 4.9, that reducing the variance of the embedding in a single-layer transformer (to less than 6% of all parameters) significantly accelerates the grokking process. This is equivalent to the positive imbalance discussed in previous sections. Therefore, this reinforces the finding that imbalance can control the learning regime in nonlinear networks.

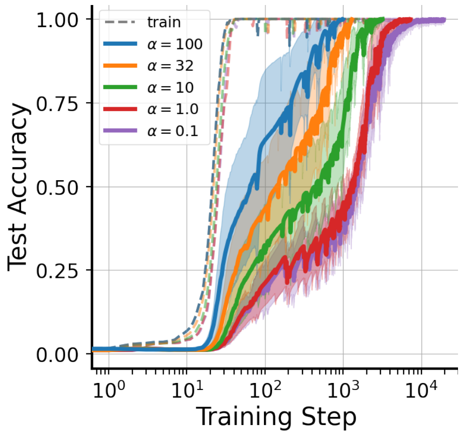
4.4.2.5 Forgetting Curves in MNIST

To further support the findings presented in Sec. 4.4.1.1 on continual learning, we now examine a continual learning task based on the MNIST dataset [61]. This dataset has previously been adapted to continual learning benchmarks, e.g. most famously in the permuted MNIST task [110]. Here, we construct a slightly different continual learning task to encode a notion of task similarity.

We begin by considering only one-half of the ten-class MNIST dataset, such that we are left with only data in the first five classes. Our first task in the sequence of two tasks consists simply of classifying these five digits. Our second task is also to classify five digits and ranges from classifying the same five digits (maximum task



(a) Convolutional Filters



(b) Transformer Grokking

Figure 4.9: Impact of balanced initialisations in practice. Here, we provide evidence that an a positively balanced initialisation - where the second layer is wider than the first (a) promotes the interpretability of early layers in CNNs and (b) decreases the time spent grokking in modular arithmetic in a transformer. In these experiments, we regulate the first layer's learning speed relative to the rest of the network by dividing its initialisation by α. For models without normalisation layers, we also scale the last layer's initialisation by α to preserve the input-output map. $\alpha = 1$ represents standard parametrisation, while $\alpha \gg 1$ and $\alpha \ll 1$ correspond to positively and negatively balanced initialisations, respectively. See Appendix B.3.9 for details.

similarity) to classifying five new digits, i.e. those that were discarded to construct the first task (orthogonal tasks- minimum task similarity). In a ten-class dataset like MNIST, this gives us only a very coarse grip on task similarity, but this suffices to robustly elicit behaviour analogous to what we observe in the toy teacher-student models.

We use a two-layer, multi-head, feed-forward architecture with sigmoidal activations to mirror the models used in the teacher-student setup. The hidden dimension of our networks needs to be larger to properly learn the classification task; we therefore lose the elegance and control afforded by the polar coordinate initialisations of Sec. 4.4.1.1 to vary entropy and scale of initialisations. The method we use to generalise this notion is to interpolate between two initialisations: a high entropy initialisation (e.g. a uniform distribution) and a relatively low entropy initialisation (e.g. Normal or Laplace distribution).

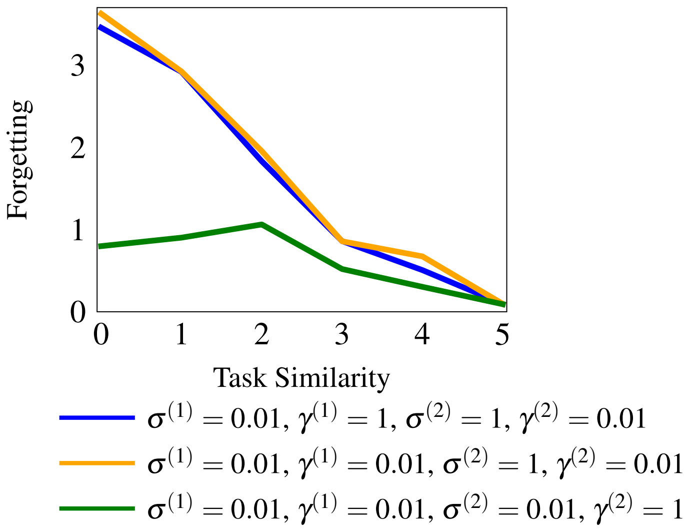


Figure 4.10: Forgetting profiles on MNIST continual learning problem. Forgetting vs. task similarity on a continual learning task using the MNIST dataset. Here, similarity is defined as the number of classes that are the same in a 5-way classification problem from the first task to the second, i.e., zero corresponds to 5 new classes and five corresponds to the same five classes. The green line is achieved by initialising with low entropy and small weights in the first head, followed by high entropy and small weights in the second, while the blue and orange lines have low entropy second head initialisations with high and low entropy initialisations in the first head, respectively. These forgetting profiles (in terms of their monotonicity patterns) qualitatively match those observed in the theoretical toy problems discussed in Sec. 4.4.1.1 (see Fig. 4.5). Note that $\sigma^{(i)}$ denotes the scale of the i^{th} head initialization (equivalent to r in Fig. 4.5) and $\gamma^{(i)}$ the relative entropy (playing a similar role to θ in Fig. 4.5). For simulation details, see Appendix B.3.10.

In Fig. 4.10, we show forgetting profiles for three different initialisation schemes (analogous to those shown in Fig. 4.5) for the continual MNIST task described above. It is clear that in the case of low entropy and specialisation in the first task, along with high entropy in the second head initialisation, we observe characteristics of Maslow's hammer. However, when we initialise the second head with low entropy, we recover the monotonic relationships found in the equivalent initialisations from the toy models. Note that, at this stage, these are primarily qualitative results, i.e. we are comparing the shapes of these forgetting profiles and not the relative magnitudes or detailed forgetting metrics. Overall, we observe similar forgetting patterns in this more complex setting, effectively validating insights from the toy model and highlighting the influence of entropy and weight imbalance.

4.5 Discussion

In this chapter, we investigate the impact of prior knowledge—modelled through initialisation—on the specialisation of representations and their learning dynamics. We introduce a minimal model that transitions between specialised and shared representations, offering a complementary perspective on both the lazy learning regime [130] and the rich learning regime [196, 233]. While our analysis is grounded in the feature learning framework, it provides a more nuanced understanding of the various rich representations that emerge as a function of initialisation. Specifically, we emphasise the impact of increasing the entropy of the readout weights and introducing asymmetries between the first and last layers to get a specialised solution. These findings extend the results from discussed in Chapter 3, while adding further nuance to the role of entropy in this mechanism. [67]. This result closely aligns with the insights derived in [158], where we further formalised the findings for two-layer ReLU networks in our paper[1]. Overall, our theory suggests that imbalanced initialisations may offer a promising strategy for optimising efficient feature learning.

However, there are several key limitations to extending this work to more general settings. One challenge is that this work primarily operates within simplified frameworks, which—while commonly used in neural network analysis— do not fully capture the complexity of modern architectures and real-world data. For example, we focus on two-layer neural networks, and applying this theory to deeper networks presents significant difficulties. In deeper architectures, which are widely used in practice, there are numerous ways to introduce imbalance across layers. In our experiment, we made informed yet arbitrary decisions to modify the balance between blocks of the network or adjust the first layer in relation to the rest of the network. Investigating how different imbalance profiles affect feature learning, inductive biases, and generalisation in deeper networks is a promising direction for future research. One potential approach is to build on the path-based framework proposed by Saxe et al. [242].

[1]This part of the paper is not discussed in detail in this thesis.

Furthermore, our experiments rely on Gaussian input data and simplified input-output relations, which are far from the complexities of real-world scenarios. A natural next step is to extend our analysis to more realistic generative models, such as the hidden manifold model [107] or the superstatistical generative model [3], which offer more structured data distributions and better capture observations from real data experiments.

Complementing analytical approaches with numerical experiments in controlled real-world settings is a promising direction to overcome these limitations. While this may reduce some of the analytical tractability, it brings us closer to solving practical challenges. For instance, we plan to explore a broader range of network architectures, datasets—such as Car3D [76] and dSprites [190]—and evaluation metrics, such as SAP [154, 120]. We also hope to deepen our practical insights on forgetting by using the continual learning benchmark proposed by [102] to test our findings in more complex environments. Although we have focused on continual learning, other domains of machine learning are also affected by specialised representations. An interesting direction for future research concerns the emergence of compositionality. [169, 71] reported the emergence of compositional representations in neural networks, and theoretical frameworks are now available to investigate this phenomenon [166]. These future studies will allow us to validate our theoretical insights and fully assess their relevance in real-world settings.

Chapter 5

Conclusion

To keep your balance, you must keep moving.

– Albert Einstein

This chapter reviews and connects all the works presented in this thesis.

5.1 Discussion

Below, we summarise and discuss the insights gained from Chapters 3 and 4 on the influence of initialisation on learning regimes. We then consider the relevance of toy models used in this work to the field of machine learning, before turning to their implications for neuroscience.

5.1.1 Balancing learning regimes

At a high level, this research investigates how prior knowledge—both in biological and artificial neural networks—influences the learning process and the formation of internal representations that support complex behaviour, including continual learning, curriculum learning, reversal learning, and the acquisition of structured knowledge. To address these questions, we developed mathematical frameworks grounded in deep learning theory, aimed at capturing the flexible and adaptive nature of learning. In our models, prior knowledge is embedded in the initial state of the network, as well as in the representations carried over from previous tasks.

Chapter 2 offers a comprehensive review of the foundational background and pertinent literature that underpin this work. We begin by outlining the key concepts of deep learning and their relevance to neuroscience. This is followed by an exploration of core themes central to the thesis, with a particular focus on the rich and lazy learning regimes, diverse learning paradigms, and the variety of neural network architectures and initialisation strategies previously studied.

In Chapter 3, we introduce a minimal model with exactly solvable learning dynamics under varying prior knowledge. This framework reveals a subtle yet critical factor: imbalance—defined as the relative scale across layers at initialisation. Although frequently overlooked, we demonstrate that relative imbalance provides access to the full spectrum of learning regimes—from rich to lazy—uncovering behaviours that could not be explained by absolute scale alone [51]. To gain a deeper understanding, we systematically vary architectural configurations, initialisation scales, and datasets and show how these elements interact with prior knowledge during the learning process.

Chapter 4 extends this investigation to non-linear networks, evaluating whether the insights gained from linear models hold in more realistic and complex settings. In particular, we underscore the importance of readout weight entropy and the asymmetry between the first and final layers at initialisation as key drivers of network specialisation—a defining characteristic of rich learning regimes (see Sec. 4.3). Moreover, we demonstrate that both relative scaling and overall weight scale critically influence the extent of feature learning in non-linear and applied settings (discussed in Sec. 4.4.2).

Overall, this thesis successfully demonstrates the impact of relative scaling on learning regimes, from theoretical linear toy models to practical neural networks. However, variations in imbalance—or relative scaling—give rise to markedly different learning behaviours depending on the activation function. For instance, in linear networks, zero imbalance typically fosters rich dynamics, whereas in ReLU-activated nonlinear networks, a positive imbalance—where the second layer has greater variance than the first—promotes the formation of structured, specialised representations. In practical models employing non-linearity, a positive imbalance likewise facilitates feature learning. Furthermore, Chapter 3 shows that network architecture itself plays a decisive role in the emergence of rich learning regimes. For instance, in linear networks, funnel-shaped architectures tend to enter a delayed rich learning regime when the relative scaling is negatively balanced, whereas anti-funnel architectures transition into this regime under negative imbalance. In our nonlinear-network experiments we exclusively employed a funnel-shaped design; consequently, a systematic exploration of alternative architectural motifs and their impact on the learning regime constitutes a promising avenue for future work.

It is worth emphasising that the two experimental setups explored in this work are not directly equivalent. The student–teacher framework presented in Chapter 4 operates in the high-dimensional limit and has long been a standard approach for probing the feature learning regime. However, its connection to the Neural Tangent Kernel (NTK) or lazy training regime is rather limited. In principle, one might attempt to study lazy dynamics by taking the limit of extreme layer imbalance. Yet, such an approach is constrained by the assumptions typically imposed in high-dimensional settings—such as Gaussian inputs and specific scaling relations. These constraints make the exploration of NTK-like behavior less relevant in this framework, particularly when compared to setups where such dynamics can be studied more flexibly and with fewer restrictions. In our setup, we deliberately avoid this regime. Even when initialising the first layer with a large scale to approximate lazy training, the network still develops representations that go beyond simple random projections. By contrast, the scenario explored in Chapter 3 involves a linear network trained on finite data, resulting in distinct training dynamics that span the full range—from lazy to rich regimes.

Furthermore, one of the persistent challenges in this area is the lack of a precise and universally accepted definition of the rich learning regime. It is often characterised in contrast to the lazy regime, where learning dynamics are governed by a fixed Neural Tangent Kernel (NTK). This regime is also often characterised by sigmoidal learning curves and simplicity biases, such as low-rankness [170] sparsity [294] or specialisation [106]. However, what constitutes "richness" in learning remains context-dependent. For example, in our linear network analysis (Chapter 3), we use the NTK framework to delineate regimes, while in the non-linear case (Chapter 4), we focus on specialisation—an indicator of rich learning—rather than relying on kernel-based descriptions.

Looking ahead, a key direction for future research is to further develop a theoretical framework that formally characterises the spectrum of learning regimes in broader settings, particularly those that fall outside the lazy regime. This thesis takes an initial step in that direction by providing a clearer mapping between lazy and rich learning, and by offering a more nuanced understanding of how different regimes emerge across architectures and initialisations. Encouragingly, several insights derived from the linear setting appear to generalise to non-linear networks in practical contexts, pointing to the broader relevance and applicability of these findings. Importantly, we show in Chapter 3 that widely used initialisation schemes can also be interpreted through the lens of balancedness. Although originally designed to address challenges such as vanishing gradients, these schemes often induce a form of balance across network layers. As demonstrated in this thesis, such balanced initialisation may implicitly steer networks towards a rich feature learning regime, potentially explaining the extrodinary ablilities of deep networks today. A systematic investigation of this phenomenon presents a promising avenue for future research.

5.1.2 The importance of toy models

In this thesis, we employ simplified or "toy" models as a means to develop theoretical understanding and build intuition in the face of the growing complexity of contemporary machine learning systems. This approach is inspired by the longstanding tradition in physics, where abstraction has been used as a powerful methodological tool to gain insight into complex natural phenomena. Such simplifications—whether modelling a cow as a sphere [139], linearising pendulum motion [127], or employing idealised frameworks such as Hopfield networks to study associative memory and network dynamics [123]—reflect a deeper epistemological principle: a well-constructed model need not capture every microscopic detail to reveal fundamental system behaviour. Instead, effective models strike a balance between analytical tractability and conceptual clarity, often producing insights that transcend their original scope of application.

In Chapter 3, we begin by analysing linear neural networks as a tractable class of models that serve as proxies for deep neural networks (DNNs), which are inherently non-linear and dynamically complex, particularly due to the presence of activation functions such as ReLU. Although often overlooked due to their apparent simplicity and frequent characterisation as mere compositions of matrix multiplications, we demonstrate that linear networks exhibit surprisingly rich dynamical properties.

Despite their linear architectures, the learning dynamics of linear neural networks are non-linear [243]. We show in Chapter 3, that these dynamics are capable of capturing a range of empirical phenomena observed in non-linear networks, including step-wise learning [243], the rich and lazy learning regimes [67], and the phenomenon of grokking [158] in algorithmic learning tasks [218] (discussed in Sec. 3.4). Similar linearised frameworks have been successfully employed to investigate a wide range of phenomena [216, 119, 6, 140], as well as learning paradigms such as self-supervised learning [258], modularisation [134], specialisation [135], fine-tuning [174], and generalisation [4]. Our primary focus in Chapter 3 is on feedforward linear models; however, we also explore recurrent architectures.

A distinctive advantage of linear network models lies in their analytical solvability under suitable assumptions. Several studies have derived exact solutions for learning trajectories in these systems [23, 243, 39], providing formal mathematical interpretations that often generalise beyond the immediate scope of the model. Motivated by this, Chapter 3 broadens the class of analytically tractable models by investigating a wider range of initialisation schemes, with particular emphasis on balanced initialisations. This extension reveals that balanced initialisation across layers plays a pivotal role in modulating the degree of feature learning. This generalisation also allows for the analysis and interpretation of more subtle dynamical behaviours, including catastrophic slowing (as examined in Sec. 3.5.2) and exponential convergence towards structured representations (see Sec. 3.4.6.1)—phenomena that are otherwise challenging to isolate in less controlled settings.

While linear models offer clarity and foundational insights, we remain cognisant of their limitations. In practical applications, linear networks are rarely employed due to their lack of expressive power and sensitivity to architectural choices. Certain behaviours inherent to DNNs, particularly those requiring activation-dependent nonlinearities, cannot be faithfully reproduced with layer-wise linear approximations. For instance, although linear models highlight the importance of balanced layers, they do not fully replicate the observed behaviours in real-world systems.

Accordingly, in Chapter 4, we turn to simplified models of non-linear networks, focusing specifically on the two-layer perceptron. This toy model architecture, grounded in statistical physics, allows for the derivation of dynamical equations governing the evolution of order parameters, albeit at the cost of analytic tractability. These equations must typically be integrated numerically. Nevertheless, this approach permits the investigation of dynamical aspects of network behaviour while incorporating critical non-linear features.Moreover, such toy models have found application in exploring cognitive capabilities such as continual learning [167], curriculum design [239], and compositionality [166].

In Chapter 4, we investigate how variations in network initialisation influence learning dynamics and representational properties. Specifically, we highlight the role of readout weight entropy and asymmetry between the first and last layers in driving network specialisation—a hallmark of rich learning regimes (discussed in Sec. 4.3). We demonstrate that the relative scaling between layers significantly modulates the degree of specialisation observed, a finding with direct relevance to practical machine learning architectures (discussed in Sec. 4.4.2). We further validate these observations in empirical settings, bridging the gap between theory and application.

In conclusion, this thesis highlights the enduring relevance of toy models—both linear and non-linear—as indispensable tools for theoretical exploration in machine learning and neuroscience. These simplified systems not only elucidate fundamental mechanisms but also inform our understanding of real-world models and their behaviours. While their full potential has yet to be realised, we advocate for the continued development and analysis of such models, which hold promise for advancing our theoretical and practical understanding of learning systems.

5.1.3 Relevance to neuroscience

Deep learning models have emerged as powerful tools for advancing our understanding of brain function, offering mechanistic insights into neural activity and cognitive processes [241]. These models provide researchers with the opportunity to test hypotheses in controlled, artificial settings and to generate predictive frameworks for biological phenomena. For instance, deep learning architectures have been widely utilised to investigate the geometry of neural representations and the principles governing information transfer [52, 53, 302, 266]. In addition, deep linear networks have yielded valuable insights into various cognitive functions, including semantic learning in infancy, task abstraction, and transitive inference in humans [247, 229, 205, 90, 87, 175]. Building on this foundation, the present thesis leverages deep learning models to explore fundamental questions in neuroscience, with particular emphasis on how prior knowledge influences subsequent learning.

In particular, the findings presented in this thesis contribute meaningfully to ongoing debates in cognitive science and neuroscience [87], particularly in relation to continual learning [121]. Our results suggest that both extrinsic factors—such as task similarity (discussed in Chapter 4) —and intrinsic elements—such as the adoption of ”rich” or ”lazy” learning strategies (discussed in Chapters 3 and 4) —play a critical role in shaping learning dynamics. This observation aligns with broader theoretical frameworks suggesting that computational principles are shared across biological and artificial systems, influenced by global structural properties and internal representations. Notably, we demonstrate that even relatively compact models are capable of producing diverse representational profiles and functional behaviours, reflecting the heterogeneity observed in human cognition.

Crucially, the models presented in this thesis are abstract and not intended as detailed simulations of biological neural circuits, they are useful for probing high-level computational principles that may be shared between artificial and biological systems. A key limitation of this approach is that it does not capture the full biophysical or anatomical complexity of the brain, nor does it produce outputs directly comparable to neural recordings without further adaptation. Additionally, the behavioural tasks used in our simulations are simplified relative to those used in experimental neuroscience. As such, the relevance to brain function should be interpreted with caution. However, by isolating core mechanisms—such as sparsity, representational overlap, and learning dynamics—these models allow us to explore hypotheses that are difficult to test directly in vivo. Their value lies in generating conceptual insights and predictions that can guide future empirical studies rather than in providing a direct one-to-one mapping with biological systems.

Despite the progress made, significant challenges remain in bridging the gap between artificial models and empirical neuroscience. A key limitation in the field is the lack of consensus regarding standardised methods for comparing model predictions with experimental data. While the importance of theory–experiment interaction is widely recognised, tools that support such integration are still in their infancy. One core issue concerns the availability and accessibility of data in consistent, labelled formats. Although there is growing momentum towards open-source collaboration, practical difficulties persist in accessing behavioural and neural datasets. Researchers are often required to interpret diverse task structures and grapple with heterogeneous pre-processing pipelines, often insufficiently documented. This lack of standardisation hinders the field's ability to comprehensively test and validate new modelling approaches. Furthermore, not all models are designed to generate neural or behavioural outputs, nor can they always interact with specific tasks. Conversely, experimental protocols are not always described in a manner that facilitates computational modelling. These disparities make it difficult to establish meaningful, systematic comparisons between models and biological data. Moreover, there is currently no streamlined or standardised method for enabling models to engage directly with behavioural tasks. Researchers frequently resort to bespoke simulations to replicate animal behaviour and environmental interactions, resulting in duplicated efforts, inconsistencies, and the increased risk of implementation errors. These barriers collectively impede efforts to make model–data integration more scalable and accessible.

Overall, this thesis underscores the importance of sustained efforts at the interface between machine learning and neuroscience, thereby advancing both disciplines through a deeper understanding of model and animal behaviour across varied experimental settings. Furthermore, we advocate for streamlining the exchange between theoretical modelling and empirical data. This integrated approach will not only expose the limitations of current models but also generate novel, testable hypotheses for future experiments.

5.2 General conclusions

Altogether, this research aims to foster a synergistic relationship between theoretical models of intelligence and practical AI applications. The outcomes of this research include the development of minimal models with exact solutions that are more aligned with realistic scenarios. The theoretical advances in architecture, initialisation, and data constraints subsequently facilitated the exploration of the factor controlling the rich and lazy regime with key applications across machine learning and neuroscience, including continual learning [212, 148, 306], reversal learning [84], transfer learning [276, 273, 161, 101], and efficient fine-tuning methods [173]. This research will open avenues for cross-domain applications through models of complex cognitive functions, along with the creation of advanced comparison tools and methods. Broadly, this work seeks to close the gaps between human and machine intelligence, paving the way for significant breakthroughs in both neuroscience and AI with the ultimate goal of enhancing our understanding of how the brain and AI systems learn and adapt.

5.3 Further work

Moving forward, building upon the foundation established in this thesis. We propose a theory-driven approach that combines theoretical insights with practical applications alongside the development of tools that enable the standardisation of representation comparisons, creating a feedback loop to guide experiments and ultimately unifying representations.

5.3.1 Theoretical models for realistic settings beyond the limitations of existing frameworks

Building on my previous work, promising research directions include analysing the learning dynamics of neural models and studying the representation of models in both functional and parameter spaces in small toy models [243, 69, 158, 39, 67]. However, current approaches remain constrained by a set of standard assumptions that limit their applicability to real-world problems. The first research proposal aims to address these limitations by critically evaluating these assumptions and introducing novel methodologies to enhance our theoretical models where necessary. We plan to explore this through three interconnected projects that complement each other.

5.3.1.1 The effect of architectural/connectivity on representation: depth and width

In the brain, distinct regions exhibit a variety of architectures, and differences in their connectivity may correspond to their specialised functional roles [162]. In this thesis, we systematically vary architectural parameters—such as depth, width and connectivity—to evaluate their effects on learning dynamics, the emergence of structured representations and overall model performance [67, 158].

Bridging analytically tractable models with real-world complexity is crucial for both machine learning and neuroscience if we are to deepen our understanding of how these systems function. Moreover, elucidating how the statistics of training data interact with architectural features to constrain the learned representations remains an open challenge—with significant implications for model learning and generalisation.

An important avenue of research lies in examining the dynamics of architectural frameworks relevant to neuroscience, such as deep feedforward, wide, and bottleneck architectures, which have been previously applied in modeling brain regions such as cerebellum-like structures [201, 295, 207, 201, 295, 207]. These architectures are regularly employed for practical applications in machine learning, for example, to implement Low-Rank adapters (LoRA) [124]. Lora architectures have become increasingly important in modern machine learning, particularly for fine-tuning large pre-trained models in natural language processing (NLP) and computer vision tasks [124, 62]. This research has the potential to theoretically explain their success and provide insights that could lead to further improvements.

By examining these architectural factors, we can make substantial progress in enhancing the characterisation, interpretability, and performance of machine learning models and theoretical models of brain regions. The ultimate goal is to study more complex architectures present in the brain and understand how circuitry constrains the types of functions it can learn.

5.3.1.2 The effect of initialisation/prior knowledge on representation: beyond Tabula rasa

A key feature of human learning is its strong dependence on prior knowledge: what we already know significantly influences how we learn new information [44]. As exemplified in this thesis, in machine learning, the role of prior knowledge is evident across several paradigms, including reversal learning [84], transfer learning [276, 273, 161, 101], continual learning [212, 148, 306], curriculum learning [30], and meta-learning [136].

Existing approaches, including our current framework [67, 39, 69], typically assume tabula rasa (small random weights that lack structure) or balanced initialisation (a property that remains conserved throughout training akin to relative scaling across layers)[12, 243]. In this thesis [69, 67, 158], we demonstrated that the degree of balance between layers plays a critical role in shaping feature learning, influencing inductive biases, and affecting generalisation capacities.

A promising direction for future research involves examining this effect within neuroscience-relevant architectures, such as deep and bottleneck networks, as discussed above. Further relaxing the constraints on initialisation could provide important insights into the generalisation, memorisation, and optimisation of artificial and biological neural networks.

5.3.1.3 The effect of the data on representation: Non-whitened/correlated inputs and realistic settings

In both machine learning and neuroscience, data encountered in practice is often high-dimensional, highly correlated, noisy, and multimodal. Analytical studies which derive exact solutions, such as the one presented here and form the foundation of existing literature [93, 39, 158] often adopt restring assumptions such as whitened input. Moving towards more realistic data settings is critical for developing models that better reflect these complexities.

Recent studies have started to relax the assumption of whitened inputs, allowing for more detailed analyses of learning dynamics on high-dimensional, realistic datasets [247, 4, 131]. Building on these advances, future work will examine the effects of input correlations and noise on learning trajectories, with the aim of elucidating the implicit biases that such realistic data conditions impose on learning processes and the resulting representational structures.This research trajectory aims to bridge theoretical models and neuroscience applications by providing insights into how neural networks process and represent complex, real-world data to support processes such as generalisation and memorisation [138, 207, 55, 38, 32].

5.3.2 Open-source comparative frameworks

In today's era of large-scale neuroscience and machine learning, a standardised framework for comparing neural representations is crucial for bridging theory and application. Building on the theoretical and application foundations established above, future research will focus on the development of software to enable more rigorous comparisons between neural representations in biological brains and computational models. By systematising and standardising model comparisons, this research direction seeks to develop and refine metrics for assessing similarity and performance. As part of this effort, *NeuralPlayground* [68], a framework designed to facilitate comparisons of neural activity in brain regions such as the hippocampus and entorhinal cortex, will continue to be developed. This includes the integration of additional datasets [27, 269, 48, 98] and the enhancement of existing models [57, 43], thereby expanding the platform's utility for both neuroscience and machine learning research communities.

Appendix A

Appendix Chapter 3

A.1 Preliminaries

A.1.1 Random weight initialisations and λ-balanced property

Throughout this work, we assume that initial weights are λ-Balanced. However, in practice, weights are not initialised with that goal in mind. Usually, a weight matrix $\mathbf{W}$ is initialised with some random distribution centred around 0, with variance inversely proportional to the number of layers on which $\mathbf{W}$ has a direct effect ([105], [164], [118]). In this section, we show that many common initialisation techniques lead to λ-Balanced weights in expectation. Furthermore, as the size of a network tends to infinity, these random weights are λ-Balanced in probability.

We do this by first finding the expectation and variance of the balance computation for two adjacent weight matrices, $\mathbf{W}_{i+1}$ and $\mathbf{W}_i$, initialised under a normal distribution with zero mean. Subsequently, we describe how network structure and size can impact the expectation and variance of the balance computation.

Theorem A.1.1. ***[Random Weight Initialisation Leads to Balanced Condition]*** *Consider a fully connected neural network with L layers. Each layer has N_i neurons, and the weights of each layer $\mathbf{W}_i$ is a matrix of dimension (N_i, N_{i+1}). The matrix $\mathbf{W}_i = (w^i_{N,m})$ where $w^i_{N,m} \sim \mathcal{N}(0, \sigma_i^2)$, where σ_i^2 is determined based on the initialisation technique. Then the following hold for all $i \in [1, L-1]$:*

- $\mathbb{E}\left[\mathbf{W}_{i+1}^T\mathbf{W}_{i+1} - \mathbf{W}_i\mathbf{W}_i^T\right] = (N_{i+2}\sigma_{i+1}^2 - N_i\sigma_i^2)\mathbf{I}$.
- *Var* $\left[\mathbf{W}_{i+1}^T\mathbf{W}_{i+1} - \mathbf{W}_i\mathbf{W}_i^T\right] = (N_{i+2}\sigma_{i+1}^4 + N_i\sigma_i^4)\mathbb{B}$, *where* $\mathbb{B}$ *is a square matrix with fours across the diagonal and ones everywhere else.*

Note that in the case $L = 3$, $N_0 = i, N_1 = h, N_2 = o$ *with* i, h, o *being the input, hidden and output dimensions respectively as defined in the main text.*

Proof of Theorem A.1.1.

$$\text{Let } \mathbf{W}_{i+1} = \begin{pmatrix} w_{1,1} & w_{1,2} & \cdots & w_{1,N_{i+2}} \\ w_{2,1} & w_{2,2} & \cdots & w_{2,N_{i+2}} \\ \vdots & \vdots & \ddots & \vdots \\ w_{N_{i+1},1} & w_{N_{i+1},2} & \cdots & w_{N_{i+1},N_{i+2}} \end{pmatrix} = \begin{pmatrix} \overline{w}_1 & \overline{w}_2 & \ldots & \overline{w}_{N_{i+2}} \end{pmatrix} \tag{A.1}$$

with $\overline{w}_j = (w_{1,j}, w_{2,j}, \ldots, w_{N_{i+1},j})^T$. Then,

$$\mathbf{W}_{i+1}^T\mathbf{W}_{i+1} = \begin{pmatrix} \overline{w}_1^T \\ \overline{w}_2^T \\ \vdots \\ \overline{w}_{N_{i+2}}^T \end{pmatrix} \begin{pmatrix} \overline{w}_1 & \overline{w}_2 & \cdots & \overline{w}_{N_{i+2}} \end{pmatrix}$$

$$= \begin{pmatrix} \langle \overline{w}_1, \overline{w}_1 \rangle & \langle \overline{w}_1, \overline{w}_2 \rangle & \cdots & \langle \overline{w}_1, \overline{w}_{N_{i+2}} \rangle \\ \langle \overline{w}_2, \overline{w}_1 \rangle & \langle \overline{w}_2, \overline{w}_2 \rangle & \cdots & \langle \overline{w}_2, \overline{w}_{N_{i+2}} \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle \overline{w}_{N_{i+2}}, \overline{w}_1 \rangle & \langle \overline{w}_{N_{i+2}}, \overline{w}_2 \rangle & \cdots & \langle \overline{w}_{N_{i+2}}, \overline{w}_{N_{i+2}} \rangle \end{pmatrix}. \tag{A.2}$$

Now, consider $\langle \overline{w}_i, \overline{w}_j \rangle$ with $i \neq j$,

$$
\begin{aligned}
\langle \overline{w}_i, \overline{w}_j \rangle &= \sum_{k=1}^{N_{i+2}} w_{k,i} w_{k,j}, && \text{(A.3)} \\
\mathbb{E}\left[\langle \overline{w}_i, \overline{w}_j \rangle\right] &= \mathbb{E}\left[\sum_{k=1}^{N_{i+2}} w_{k,i} w_{k,j}\right] \\
&= \sum_{k=1}^{N_{i+2}} \mathbb{E}[w_{k,i} w_{k,j}] \\
&= \sum_{k=1}^{N_{i+2}} \mathbb{E}[w_{k,i}]\mathbb{E}[w_{k,j}] = 0 \quad \text{(by independence)}, && \text{(A.4)} \\
\operatorname{Var}\left[\langle \overline{w}_i, \overline{w}_j \rangle\right] &= \mathbb{E}\left[\langle \overline{w}_i, \overline{w}_j \rangle^2\right] - \left[\mathbb{E}\left[\langle \overline{w}_i, \overline{w}_j \rangle\right]\right]^2 \\
&= \mathbb{E}\left[\left(\sum_{k=1}^{N_{i+2}} w_{k,i} w_{k,j}\right)^2\right] \\
&= \mathbb{E}\left[\sum_{k=1}^{N_{i+2}} w_{k,i}^2 w_{k,j}^2 + 2\sum_{k=1}^{N_{i+2}} \sum_{l>k} w_{k,i} w_{k,j} w_{l,i} w_{l,j}\right] \\
&= \sum_{k=1}^{N_{i+2}} \mathbb{E}[w_{k,i}^2 w_{k,j}^2] + 2\sum_{k=1}^{N_{i+2}} \sum_{l>k} \mathbb{E}[w_{k,i}]\mathbb{E}[w_{k,j}]\mathbb{E}[w_{l,i}]\mathbb{E}[w_{l,j}] \\
&= \sum_{k=1}^{N_{i+2}} \mathbb{E}[w_{k,i}^2]\mathbb{E}[w_{k,j}^2] \\
&= (N_{i+2})\sigma_{i+1}^4. && \text{(A.5)}
\end{aligned}
$$

Similarly, consider $\langle \overline{w}_i, \overline{w}_i \rangle$:

$$\langle \overline{w}_i, \overline{w}_i \rangle = \sum_{k=1}^{N_{i+2}} w_{k,i}^2. \tag{A.6}$$

$$\mathbb{E}\left[\langle \overline{w}_i, \overline{w}_i \rangle\right] = \mathbb{E}\left[\sum_{k=1}^{N_{i+2}} w_{k,i}^2\right] = N_{i+2}\mathbb{E}\left[w_{k,i}^2\right] = N_{i+2}\sigma_{N_{i+1}}^2. \tag{A.7}$$

$$\begin{aligned}
\operatorname{Var}\left[\langle\overline{w}_i,\overline{w}_i\rangle\right] &= \mathbb{E}\left[\left(\langle\overline{w}_i,\overline{w}_i\rangle\right)^2\right] - \mathbb{E}\left[\langle\overline{w}_i,\overline{w}_i\rangle\right]^2 \\
&= \mathbb{E}\left[\left(\sum_{k=1}^{N_{i+2}} w_{k,i}^2\right)^2\right] - N_{i+2}^2\sigma_{N_{i+1}}^4 \\
&= \mathbb{E}\left[\left(\sum_{k=1}^{N_{i+2}} w_{k,i}^2\right)^2\right] - N_{i+2}^2\sigma_{N_{i+1}}^4 \\
&= \mathbb{E}\left[\sum_{k=1}^{N_{i+2}} w_{k,i}^4 + 2\sum_{k=1}^{N_{i+2}}\sum_{l=k+1}^{N_{i+2}} w_{k,i}^2 w_{l,i}^2\right] - N_{i+2}^2\sigma_{N_{i+1}}^4 \\
&= \sum_{k=1}^{N_{i+2}} \mathbb{E}\left[w_{k,i}^4\right] + 2\sum_{k=1}^{N_{i+2}}\sum_{l=k+1}^{N_{i+2}} \mathbb{E}\left[w_{k,i}^2\right]\mathbb{E}\left[w_{l,i}^2\right] - N_{i+2}^2\sigma_{N_{i+1}}^4 \\
&= N_{i+2}(3\sigma_{N_{i+1}}^4) + (N_{i+2}^2 - N_{i+2})\sigma_{N_{i+1}}^4 - N_{i+2}^2\sigma_{N_{i+1}}^4 \\
&= 4N_{i+2}\sigma_{N_{i+1}}^4. \qquad (A.8)
\end{aligned}$$

Hence

$$\mathbb{E}\left[\mathbf{W}_{i+1}^T\mathbf{W}_{i+1}\right] = \left(N_{i+2}\sigma_{i+1}^2\right)\mathbf{I}, \qquad (A.9)$$

$$\operatorname{Var}\left[\mathbf{W}_{i+1}^T\mathbf{W}_{i+1}\right] = \left(N_{i+2}\right)\sigma_{i+1}^4\mathbb{B}. \qquad (A.10)$$

For the case for $\mathbf{W}_i$, notice we can express $\mathbf{W}_i\mathbf{W}_i^T$ as $(\mathbf{W}_i^T)^T(\mathbf{W}_\mathbf{i}^\mathbf{T})$. Hence we can use the proof above, with $\mathbf{W}_{i+1}' = \mathbf{W}_i^T$. In this case the matrix $\mathbf{W}_{i+1}'$ has shape (N_i, N_{i+1}), and each element of the matrix has variance σ_i^2. We have

$$\mathbb{E}\left[\mathbf{W}_i\mathbf{W}_i^T\right] = N_i\sigma_i^2\mathbf{I} \qquad \text{and} \qquad \operatorname{Var}\left[\mathbf{W}_i\mathbf{W}_i^T\right] = N_i\sigma_i^4\mathbb{B}. \qquad (A.11)$$

By assumption, $\mathbf{W}_i, \mathbf{W}_{i+1}$ are independent. Hence $Cov(\mathbf{W}_i, \mathbf{W}_{i+1}) = 0$. We can use this property together with linearity of expectation:

$$\mathbb{E}\left[\mathbf{W}_{i+1}^T\mathbf{W}_{i+1} - \mathbf{W}_i^T\mathbf{W}_i\right] = \left(N_{i+2}\sigma_{i+1}^2 - N_i\sigma_i^2\right)\mathbf{I}. \qquad (A.12)$$

$$\mathrm{Var}\left[\mathbf{W}_{i+1}^T\mathbf{W}_{i+1} - \mathbf{W}_i^T\mathbf{W}_i\right] = \left(N_{i+2}\sigma_{i+1}^4 + N_i\sigma_i^4\right)\mathbb{B}. \tag{A.13}$$

This completes the proof. □

In neural network training, proper weight initialisation is crucial for ensuring stable gradients during backpropagation, which helps to avoid issues such as vanishing and exploding gradients. The goal of weight scaling is to maintain appropriate variance across layers, enabling efficient and effective learning ([105]). The weights are typically initialised to be random and centered around 0 to break symmetry and ensure that different neurons learn different features.

Some of the most commonly used initialisation methods are detailed below:

- **LeCun Initialisation [164]**: Weights are initialised using a normal distribution with a mean of 0 and a variance of $\frac{1}{N_i}$, where N_i is the number of input units in the layer. Mathematically, the weights w are drawn from $\mathcal{N}(0, \frac{1}{N_i})$.
- **Glorot Initialisation [105]**: Weights are initialised using a normal distribution with a mean of 0 and a variance of $\frac{2}{N_i+N_{i+1}}$, where N_i is the number of input units and N_{i+1} is the number of output units. This method balances the variance between layers with different widths. Mathematically, the weights w are drawn from $\mathcal{N}(0, \frac{2}{N_i+N_{i+1}})$.
- **He Initialisation [118]**: Weights are initialised using a normal distribution with a mean of 0 and a variance of $\frac{2}{N_i}$, where N_i is the number of input units in the layer. This method is particularly suited for layers with ReLU activation functions. Mathematically, the weights w are drawn from $\mathcal{N}(0, \frac{2}{N_i})$.
- **Scaled Initialisation [223]**: Weights are initialised using a normal distribution with a mean of 0 and a variance of $\frac{\alpha_i}{N_i}$, where N_i is the number of input units in the layer and α_i is a parameter specific to each layer. Mathematically, the weights w are drawn from $\mathcal{N}(0, \frac{\alpha_i}{N_i})$.

These initialisation methods help ensure that the network starts with weights that facilitate stable and efficient learning, avoiding the common pitfalls of poorly initialised neural networks. Using (Theorem A.1.1), we can calculate the respective expectations and variances of the balanced computation under the different initialisations.

Initialisation	$\mathrm{Var}(w^{i+1}_{N,m})$	$\mathrm{Var}(w^{i}_{N,m})$	$\mathbb{E}[\mathbf{W}^T_{i+1}\mathbf{W}_{i+1} - \mathbf{W}_i\mathbf{W}^T_i]$	$\mathrm{Var}[\mathbf{W}^T_{i+1}\mathbf{W}_{i+1} - \mathbf{W}_i\mathbf{W}^T_i]$
LeCun	$\frac{1}{N_{i+1}}$	$\frac{1}{N_i}$	$\left(\frac{N_{i+2}}{N_{i+1}} - 1\right)\mathbf{I}$	$\left(\frac{N_{i+2}}{N^2_{i+1}} + \frac{1}{N_i}\right)\mathbb{B}$
Glorot	$\frac{2}{N_{i+1}+N_{i+2}}$	$\frac{2}{N_{i+1}+N_i}$	$2\left(\frac{N_{i+2}}{N_{i+1}+N_{i+2}} - \frac{N_i}{N_{i+1}+N_i}\right)\mathbf{I}$	$(N_{i+2}\left(\frac{2}{N_{i+1}+N_{i+2}}\right)^2 + N_i\left(\frac{2}{N_i+N_{i+1}}\right)^2)\mathbb{B}$
He	$\frac{2}{N_{i+1}}$	$\frac{2}{N_i}$	$2\left(\frac{N_{i+2}}{N_{i+1}} - 1\right)\mathbf{I}$	$4\left(\frac{N_{i+2}}{N^2_{i+1}} + \frac{1}{N_i}\right)\mathbb{B}$
Scaled	$\frac{\alpha^2_{i+1}}{N_{i+1}}$	$\frac{\alpha^2_i}{N_i}$	$\left(\frac{N_{i+2}}{N_{i+1}}\alpha^2_{i+1} - \alpha^2_i\right)\mathbf{I}$	$(N_{i+2}\left(\frac{\alpha^2_{i+1}}{N_{i+1}}\right)^2 + N_i\left(\frac{\alpha^2_i}{N_i}\right)^2)\mathbb{B}$

Table A.1: Comparison of variance and expectation of balanced computation for different weight initialisations

Table A.1 shows that for the above initialisations the balanced computation of the weight pair will result in λ-Balanced weights for some λ. Fig. A.1 also details how network structure will influence the value of λ for different initialisation techniques. These findings provide motivation to better understand the relation between the balanced computation of a network, its structure and the regime it will learn in. If we are able to understand the relation between λ-Balanced weights and rich and lazy learning in linear Networks, one might be able to approximate these results to the nonlinear case. A possible future application might be the ability to heavily influence a network's learning regime by altering the relative width of its layers, its activation functions or weight initialisation techniques used for each layer.

A.1.2 Fukumizu approach

Lemma A.1.2. *We introduce the variables*

$$\mathbf{Q} = \begin{bmatrix} \mathbf{W}^T_1 \\ \mathbf{W}_2 \end{bmatrix} \quad \textit{and} \quad \mathbf{Q}\mathbf{Q}^T = \begin{bmatrix} \mathbf{W}^T_1\mathbf{W}_1 & \mathbf{W}^T_1\mathbf{W}^T_2 \\ \mathbf{W}_2\mathbf{W}_1 & \mathbf{W}_2\mathbf{W}^T_2 \end{bmatrix}. \tag{A.14}$$

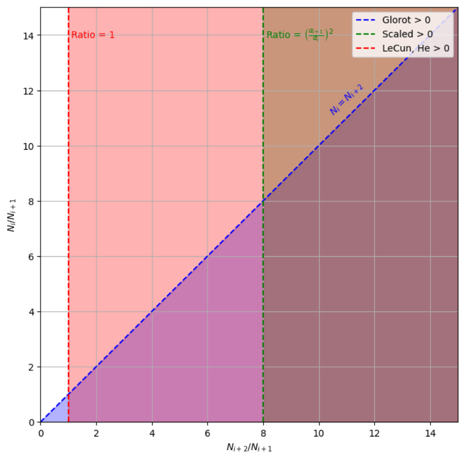


Figure A.1: Impact of Network Structure on sign of balanced coefficient for different initialisations. Lecun, He and Scaled initialisations depend solely on the ratio of the sizes of the output and hidden layers. Scaled initialisation's dependence is affected by the parameters α_{i+1}, α_i. Glorot initialisation's sign depends on both ratios.

Defining

$$\mathbf{F} = \begin{bmatrix} -\frac{\lambda}{2}I & (\tilde{\mathbf{\Sigma}}^{yx})^T \\ \tilde{\mathbf{\Sigma}}^{yx} & \frac{\lambda}{2}I \end{bmatrix}, \tag{A.15}$$

the gradient flow dynamics of $\mathbf{Q}\mathbf{Q}^T(t)$ *can be written as a differential matrix Riccati equation*

$$\tau\frac{d}{dt}(\mathbf{Q}\mathbf{Q}^T) = \mathbf{F}\mathbf{Q}\mathbf{Q}^T + \mathbf{Q}\mathbf{Q}^T\mathbf{F} - (\mathbf{Q}\mathbf{Q}^T)^2. \tag{A.16}$$

Proof. We introduce the variables

$$\mathbf{Q} = \begin{bmatrix} \mathbf{W}_1^T \\ \mathbf{W}_2 \end{bmatrix} \quad \text{and} \quad \mathbf{Q}\mathbf{Q}^T = \begin{bmatrix} \mathbf{W}_1^T\mathbf{W}_1 & \mathbf{W}_1^T\mathbf{W}_2^T \\ \mathbf{W}_2\mathbf{W}_1 & \mathbf{W}_2\mathbf{W}_2^T \end{bmatrix}. \tag{A.17}$$

We compute the time derivative

$$\tau\frac{d}{dt}(\mathbf{Q}\mathbf{Q}^T) = \tau\begin{bmatrix} \frac{d\mathbf{W}_1^T}{dt}\mathbf{W}_1 + \mathbf{W}_1^T\frac{d\mathbf{W}_1}{dt} & \frac{d\mathbf{W}_1^T}{dt}\mathbf{W}_2 + \mathbf{W}_1^T\frac{d\mathbf{W}_2}{dt} \\ \frac{d\mathbf{W}_2}{dt}\mathbf{W}_1 + \mathbf{W}_2\frac{d\mathbf{W}_1}{dt} & \frac{d\mathbf{W}_2^T}{dt}\mathbf{W}_2 + \mathbf{W}_2^T\frac{d\mathbf{W}_2}{dt} \end{bmatrix}. \tag{A.18}$$

Using equations 18 and 19, we compute the four quadrants separately giving

$$\tau\left(\frac{d\mathbf{W}_1^T}{dt}\mathbf{W}_1 + \mathbf{W}_1^T\frac{d\mathbf{W}_1}{dt}\right) =$$

$$= (\Sigma^{yx} - \mathbf{W}_2\mathbf{W}_1)^T\mathbf{W}_2\mathbf{W}_1 + \mathbf{W}_1^T\mathbf{W}_2^T(\Sigma^{yx} - \mathbf{W}_2\mathbf{W}_1)$$

$$= (\Sigma^{yx})^T\mathbf{W}_2\mathbf{W}_1 + \mathbf{W}_1^T\mathbf{W}_2^T\Sigma^{yx} - \mathbf{W}_1^T\mathbf{W}_2^T\mathbf{W}_2\mathbf{W}_1 - (\mathbf{W}_2\mathbf{W}_1)^T\mathbf{W}_2\mathbf{W}_1$$

$$= (\Sigma^{yx})^T\mathbf{W}_2\mathbf{W}_1 + \mathbf{W}_1^T\mathbf{W}_2^T\Sigma^{yx} - \mathbf{W}_1^T\mathbf{W}_2^T\mathbf{W}_2\mathbf{W}_1 - \mathbf{W}_1^T\mathbf{W}_1\mathbf{W}_1^T\mathbf{W}_1 - \lambda\mathbf{W}_1^T\mathbf{W}_1, \tag{A.19}$$

$$\tau\left(\frac{d\mathbf{W}_1^T}{dt}\mathbf{W}_2^T + \mathbf{W}_1^T\frac{d\mathbf{W}_2^T}{dt}\right) =$$

$$= (\Sigma^{yx} - \mathbf{W}_2\mathbf{W}_1)^T\mathbf{W}_2\mathbf{W}_2^T + \mathbf{W}_1^T\mathbf{W}_1(\Sigma^{yx} - \mathbf{W}_2\mathbf{W}_1)^T$$

$$= (\Sigma^{yx})^T\mathbf{W}_2\mathbf{W}_2^T + \mathbf{W}_1^T\mathbf{W}_1(\Sigma^{yx})^T - \mathbf{W}_1^T\mathbf{W}_1\mathbf{W}_1^T\mathbf{W}_2^T - \mathbf{W}_1^T\mathbf{W}_2^T\mathbf{W}_2\mathbf{W}_2^T, \tag{A.20}$$

$$\tau\left(\frac{d\mathbf{W}_2}{dt}\mathbf{W}_1 + \mathbf{W}_2\frac{d\mathbf{W}_1}{dt}\right) =$$

$$= (\Sigma^{yx} - \mathbf{W}_2\mathbf{W}_1)\mathbf{W}_1^T\mathbf{W}_1 + \mathbf{W}_2\mathbf{W}_2^T(\Sigma^{yx} - \mathbf{W}_2\mathbf{W}_1)$$

$$= \Sigma^{yx}\mathbf{W}_1^T\mathbf{W}_1 + \mathbf{W}_2\mathbf{W}_2^T\Sigma^{yx} - \mathbf{W}_2\mathbf{W}_2^T\mathbf{W}_2\mathbf{W}_1 - \mathbf{W}_2\mathbf{W}_1\mathbf{W}_1^T\mathbf{W}_1, \tag{A.21}$$

$$\tau\left(\frac{d\mathbf{W}_2}{dt}\mathbf{W}_2^T + \mathbf{W}_2\frac{d\mathbf{W}_2^T}{dt}\right) =$$

$$(\tilde{\mathbf{\Sigma}}^{yx} - \mathbf{W}_2\mathbf{W}_1)\mathbf{W}_1^T\mathbf{W}_2^T + \mathbf{W}_2\mathbf{W}_1(\tilde{\mathbf{\Sigma}}^{yx} - \mathbf{W}_2\mathbf{W}_1)^T$$

$$= \tilde{\mathbf{\Sigma}}^{yx}\mathbf{W}_1^T\mathbf{W}_2^T + \mathbf{W}_2\mathbf{W}_1(\tilde{\mathbf{\Sigma}}^{yx})^T - \mathbf{W}_2\mathbf{W}_1\mathbf{W}_1^T\mathbf{W}_2^T - \mathbf{W}_2\mathbf{W}_1(\mathbf{W}_2\mathbf{W}_1)^T$$

$$= \tilde{\mathbf{\Sigma}}^{yx}\mathbf{W}_1^T\mathbf{W}_2^T + \mathbf{W}_2\mathbf{W}_1(\tilde{\mathbf{\Sigma}}^{yx})^T - \mathbf{W}_2\mathbf{W}_1\mathbf{W}_1^T\mathbf{W}_2^T - \mathbf{W}_2\mathbf{W}_1\mathbf{W}_1^T\mathbf{W}_2^T$$

$$= \tilde{\mathbf{\Sigma}}^{yx}\mathbf{W}_1^T\mathbf{W}_2^T + \mathbf{W}_2\mathbf{W}_1(\tilde{\mathbf{\Sigma}}^{yx})^T - \mathbf{W}_2\mathbf{W}_1\mathbf{W}_1^T\mathbf{W}_2^T - \mathbf{W}_2\mathbf{W}_2^T\mathbf{W}_2\mathbf{W}_2^T + \lambda\mathbf{W}_2\mathbf{W}_2^T. \tag{A.22}$$

Defining

$$\mathbf{F} = \begin{bmatrix} -\frac{\lambda}{2}I & (\tilde{\mathbf{\Sigma}}^{yx})^T \\ \tilde{\mathbf{\Sigma}}^{yx} & \frac{\lambda}{2}I \end{bmatrix}, \tag{A.23}$$

the gradient flow dynamics of $\mathbf{QQ}^T(t)$ can be written as a differential matrix Riccati equation

$$\tau \frac{d}{dt}(\mathbf{Q}\mathbf{Q}^T) = \mathbf{F}\mathbf{Q}\mathbf{Q}^T + \mathbf{Q}\mathbf{Q}^T\mathbf{F} - (\mathbf{Q}\mathbf{Q}^T)^2. \tag{A.24}$$

We write $\tau\frac{d}{dt}(\mathbf{Q}\mathbf{Q}^T)$ for completeness

$$\begin{aligned}
&\tau \frac{d}{dt}(\mathbf{Q}\mathbf{Q}^T) = \\
&\begin{bmatrix} -\frac{\lambda}{2} & (\tilde{\mathbf{\Sigma}}^{yx})^T \\ \tilde{\mathbf{\Sigma}}^{yx} & \frac{\lambda}{2} \end{bmatrix} \begin{bmatrix} \mathbf{W}_1^T\mathbf{W}_1 & \mathbf{W}_1^T\mathbf{W}_2 \\ \mathbf{W}_2\mathbf{W}_1 & \mathbf{W}_2\mathbf{W}_2^T \end{bmatrix} + \begin{bmatrix} \mathbf{W}_1^T\mathbf{W}_1 & \mathbf{W}_1^T\mathbf{W}_2 \\ \mathbf{W}_2\mathbf{W}_1 & \mathbf{W}_2\mathbf{W}_2^T \end{bmatrix}^T \begin{bmatrix} -\frac{\lambda}{2} & (\tilde{\mathbf{\Sigma}}^{yx})^T \\ \tilde{\mathbf{\Sigma}}^{yx} & \frac{\lambda}{2} \end{bmatrix} \\
&- \begin{bmatrix} \mathbf{W}_1^T\mathbf{W}_1 & \mathbf{W}_1^T\mathbf{W}_2 \\ \mathbf{W}_2\mathbf{W}_1 & \mathbf{W}_2\mathbf{W}_2^T \end{bmatrix}^2 \\
&= \begin{bmatrix} -\frac{\lambda}{2} & (\tilde{\mathbf{\Sigma}}^{yx})^T \\ \tilde{\mathbf{\Sigma}}^{yx} & \frac{\lambda}{2} \end{bmatrix} \begin{bmatrix} \mathbf{W}_1^T\mathbf{W}_1 & \mathbf{W}_1^T\mathbf{W}_2 \\ \mathbf{W}_2\mathbf{W}_1 & \mathbf{W}_2\mathbf{W}_2^T \end{bmatrix} + \begin{bmatrix} \mathbf{W}_1^T\mathbf{W}_1 & \mathbf{W}_1^T\mathbf{W}_2 \\ \mathbf{W}_2\mathbf{W}_1 & \mathbf{W}_2\mathbf{W}_2^T \end{bmatrix}^T \begin{bmatrix} -\frac{\lambda}{2} & (\tilde{\mathbf{\Sigma}}^{yx})^T \\ \tilde{\mathbf{\Sigma}}^{yx} & \frac{\lambda}{2} \end{bmatrix} \\
&- \begin{bmatrix} \mathbf{W}_1^T\mathbf{W}_1 & \mathbf{W}_1^T\mathbf{W}_2 \\ \mathbf{W}_2\mathbf{W}_1 & \mathbf{W}_2\mathbf{W}_2^T \end{bmatrix} \begin{bmatrix} \mathbf{W}_1^T\mathbf{W}_1 & \mathbf{W}_1^T\mathbf{W}_2 \\ \mathbf{W}_2\mathbf{W}_1 & \mathbf{W}_2\mathbf{W}_2^T \end{bmatrix} \\
&= \begin{bmatrix} -\frac{\lambda}{2}\mathbf{W}_1^T\mathbf{W}_1 + (\tilde{\mathbf{\Sigma}}^{yx})^T\mathbf{W}_2\mathbf{W}_1 & -\frac{\lambda}{2}\mathbf{W}_1^T\mathbf{W}_2 + (\tilde{\mathbf{\Sigma}}^{yx})^T\mathbf{W}_2\mathbf{W}_2^T \\ \tilde{\mathbf{\Sigma}}^{yx}\mathbf{W}_1^T\mathbf{W}_1 + \frac{\lambda}{2}\mathbf{W}_2\mathbf{W}_1 & \tilde{\mathbf{\Sigma}}^{yx}\mathbf{W}_1^T\mathbf{W}_2^T + \frac{\lambda}{2}\mathbf{W}_2\mathbf{W}_2^T \end{bmatrix} . \\
&+ \begin{bmatrix} -\frac{\lambda}{2}\mathbf{W}_1^T\mathbf{W}_1 + \mathbf{W}_1^T\mathbf{W}_1(\tilde{\mathbf{\Sigma}}^{yx})^T & \frac{\lambda}{2}\mathbf{W}_1^T\mathbf{W}_2 + \mathbf{W}_1^T\mathbf{W}_2(\tilde{\mathbf{\Sigma}}^{yx})^T \\ -\frac{\lambda}{2}\mathbf{W}_2^T\mathbf{W}_1 + \mathbf{W}_2\mathbf{W}_1(\tilde{\mathbf{\Sigma}}^{yx})^T & \frac{\lambda}{2}\mathbf{W}_2\mathbf{W}_2^T + \mathbf{W}_2\mathbf{W}_2^T(\tilde{\mathbf{\Sigma}}^{yx})^T \end{bmatrix} \\
&- \begin{bmatrix} \mathbf{W}_1^T\mathbf{W}_1 & \mathbf{W}_1^T\mathbf{W}_2 \\ \mathbf{W}_2\mathbf{W}_1 & \mathbf{W}_2\mathbf{W}_2^T \end{bmatrix} \begin{bmatrix} \mathbf{W}_1^T\mathbf{W}_1 & \mathbf{W}_1^T\mathbf{W}_2 \\ \mathbf{W}_2\mathbf{W}_1 & \mathbf{W}_2\mathbf{W}_2^T \end{bmatrix} \\
&= \begin{bmatrix} -\frac{\lambda}{2}\mathbf{W}_1^T\mathbf{W}_1 + (\tilde{\mathbf{\Sigma}}^{yx})^T\mathbf{W}_2\mathbf{W}_1 & -\frac{\lambda}{2}\mathbf{W}_1^T\mathbf{W}_2 + (\tilde{\mathbf{\Sigma}}^{yx})^T\mathbf{W}_2\mathbf{W}_2^T \\ \tilde{\mathbf{\Sigma}}^{yx}\mathbf{W}_1^T\mathbf{W}_1 + \frac{\lambda}{2}\mathbf{W}_2\mathbf{W}_1 & \tilde{\mathbf{\Sigma}}^{yx}\mathbf{W}_1^T\mathbf{W}_2^T + \frac{\lambda}{2}\mathbf{W}_2\mathbf{W}_2^T \end{bmatrix} \\
&+ \begin{bmatrix} -\frac{\lambda}{2}\mathbf{W}_1^T\mathbf{W}_1 + \mathbf{W}_1^T\mathbf{W}_1(\tilde{\mathbf{\Sigma}}^{yx})^T & \frac{\lambda}{2}\mathbf{W}_1^T\mathbf{W}_2 + \mathbf{W}_1^T\mathbf{W}_2(\tilde{\mathbf{\Sigma}}^{yx})^T \\ -\frac{\lambda}{2}\mathbf{W}_2^T\mathbf{W}_1 + \mathbf{W}_2\mathbf{W}_1(\tilde{\mathbf{\Sigma}}^{yx})^T & \frac{\lambda}{2}\mathbf{W}_2\mathbf{W}_2^T + \mathbf{W}_2\mathbf{W}_2^T(\tilde{\mathbf{\Sigma}}^{yx})^T \end{bmatrix} \\
&- \begin{bmatrix} \mathbf{W}_1^T\mathbf{W}_1\mathbf{W}_1^T\mathbf{W}_1 + \mathbf{W}_1^T\mathbf{W}_2\mathbf{W}_2^T\mathbf{W}_1 & \mathbf{W}_1^T\mathbf{W}_1\mathbf{W}_1^T\mathbf{W}_2 + \mathbf{W}_1^T\mathbf{W}_2\mathbf{W}_2^T\mathbf{W}_2 \\ \mathbf{W}_2\mathbf{W}_1\mathbf{W}_1^T\mathbf{W}_1 + \mathbf{W}_2\mathbf{W}_2^T\mathbf{W}_2\mathbf{W}_1 & \mathbf{W}_2\mathbf{W}_1\mathbf{W}_1^T\mathbf{W}_2 + \mathbf{W}_2\mathbf{W}_2^T\mathbf{W}_2\mathbf{W}_2^T \end{bmatrix}
\end{aligned}$$

The four quadrants of Eq. A.18 are equivalent to equations A.19, A.20, A.21, and A.22 respectively. □

A.1.3 Representational similarity analysis and finite-width neural tangent kernel

The task-relevant representational similarity matrix [151] of the hidden layer, calculated from the inputs $\mathbf{W}_1\mathbf{X}$ is

$$\begin{aligned}\mathrm{RSM}_I(t) &= (\mathbf{W}_1(t)\mathbf{X})^T\mathbf{W}_1(t)\mathbf{X} \\ &= \mathbf{X}^T(\mathbf{W}_1^T\mathbf{W}_1)(t)\mathbf{X}.\end{aligned} \tag{A.25}$$

Similarly, the representational similarity matrix of the hidden layer, calculated from the outputs $\mathbf{W}_2^+\mathbf{Y}$, where $+$ denotes the pseudoinverse, is

$$\begin{aligned}\mathrm{RSM}_O(t) &= (\mathbf{W}_2^+(t)\mathbf{Y})^T\mathbf{W}_2^+(t)\mathbf{Y} \\ &= \mathbf{Y}^T(\mathbf{W}_2\mathbf{W}_2^T(t))^+\mathbf{Y}.\end{aligned} \tag{A.26}$$

In the following, we derive the finite-width neural tangent kernel [130] for a two-layer linear network. Starting with the network function at time t

$$F_t(\mathbf{X}) = \mathbf{W}_2\mathbf{W}_1\mathbf{X}, \tag{A.27}$$

the discrete time gradient descent dynamics of the next time step yields

$$\begin{aligned}F_{t+1}(\mathbf{X}) &= \left(\mathbf{W}_2 - \eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_2}\right)\left(\mathbf{W}_1 - \eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_1}\right)\mathbf{X} \\ &= \mathbf{W}_2\mathbf{W}_1\mathbf{X} - \eta\left(\mathbf{W}_2\frac{\partial\mathcal{L}}{\partial\mathbf{W}_1} + \frac{\partial\mathcal{L}}{\partial\mathbf{W}_2}\mathbf{W}_1 - \eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_2}\frac{\partial\mathcal{L}}{\partial\mathbf{W}_1}\right)\mathbf{X}.\end{aligned} \tag{A.28}$$

The network function's gradient flow can then be derived as

$$\frac{F_{t+1}(\mathbf{X}) - F_t(\mathbf{X})}{\eta} = -\left(\mathbf{W}_2\frac{\partial\mathcal{L}}{\partial\mathbf{W}_1} + \frac{\partial\mathcal{L}}{\partial\mathbf{W}_2}\mathbf{W}_1 - \eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_2}\frac{\partial\mathcal{L}}{\partial\mathbf{W}_1}\right)\mathbf{X} \tag{A.29}$$

$$\xrightarrow[\eta\to 0]{} \frac{d}{dt}F(\mathbf{X}) = -\left(\mathbf{W}_2\frac{\partial\mathcal{L}}{\partial\mathbf{W}_1} + \frac{\partial\mathcal{L}}{\partial\mathbf{W}_2}\mathbf{W}_1\right)\mathbf{X}. \tag{A.30}$$

Substituting the partial derivatives

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}_1} = \frac{1}{2}\frac{\partial}{\partial \mathbf{W}_1}||\mathbf{W}_2\mathbf{W}_1\mathbf{X} - \mathbf{Y}||_F^2$$
$$= \mathbf{W}_2^T(\mathbf{W}_2\mathbf{W}_1\mathbf{X} - \mathbf{Y})\mathbf{X}^T \tag{A.31}$$

and

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}_2} = \frac{1}{2}\frac{\partial}{\partial \mathbf{W}_2}||\mathbf{W}_2\mathbf{W}_1\mathbf{X} - \mathbf{Y}||_F^2$$
$$= (\mathbf{W}_2\mathbf{W}_1\mathbf{X} - \mathbf{Y})\mathbf{X}^T\mathbf{W}_1^T \tag{A.32}$$

then yields

$$\frac{d}{dt}F(X) = -\mathbf{W}_2\mathbf{W}_2^T(\mathbf{W}_2\mathbf{W}_1\mathbf{X} - \mathbf{Y})\mathbf{X}^T\mathbf{X} - (\mathbf{W}_2\mathbf{W}_1\mathbf{X} - \mathbf{Y})\mathbf{X}^T\mathbf{W}_1^T\mathbf{W}_1\mathbf{X}. \tag{A.33}$$

Finally, we introduce the identity matrix $\mathbf{I}_{N_o}$ of size N_o and apply row-wise vectoriasation $\text{vec}_\text{r}(F(\mathbf{X})) := f(\mathbf{X})$ and the identity $\text{vec}_\text{r}(\mathbf{ABC}) = (\mathbf{A} \otimes \mathbf{C}^T)\,\text{vec}_\text{r}(\mathbf{B})$ to derive the neural tangent kernel

$$\frac{d}{dt}F(\mathbf{X}) = -\mathbf{W}_2\mathbf{W}_2^T(\mathbf{W}_2\mathbf{W}_1\mathbf{X} - \mathbf{Y})\mathbf{X}^T\mathbf{X} - \mathbf{I}_{N_o}(\mathbf{W}_2\mathbf{W}_1\mathbf{X} - \mathbf{Y})\mathbf{X}^T\mathbf{W}_1^T\mathbf{W}_1\mathbf{X} \tag{A.34}$$

$$\begin{aligned}
\Leftrightarrow \frac{d}{dt}f(\mathbf{X}) &= -\left(\underbrace{\mathbf{W}_2\mathbf{W}_2^T \otimes \mathbf{X}^T\mathbf{X} + \mathbf{I} \otimes \mathbf{X}^T\mathbf{W}_1^T\mathbf{W}_1\mathbf{X}}_{\text{NTK}}\right)\text{vec}_\text{r}(\mathbf{W}_2\mathbf{W}_1\mathbf{X} - \mathbf{Y}) \\
&= -\left(\left[\mathbf{W}_2 \otimes \mathbf{X}^T, \mathbf{I} \otimes \mathbf{X}^T\mathbf{W}_1^T\right]\left[\mathbf{W}_2 \otimes \mathbf{X}^T, \mathbf{I} \otimes \mathbf{X}^T\mathbf{W}_1^T\right]^T\right)\text{vec}_\text{r}\left(\frac{\partial \mathcal{L}}{\partial F}\right) \\
&= -\left([\nabla_{\mathbf{W}_1}f, \nabla_{\mathbf{W}_2}f][\nabla_{\mathbf{W}_1}f, \nabla_{\mathbf{W}_2}f]^T\right)\frac{\partial \mathcal{L}}{\partial f} \\
&= -\left(\nabla_\theta f \nabla_\theta f^T\right)\frac{\partial \mathcal{L}}{\partial f},
\end{aligned} \tag{A.35}$$

where $[\mathbf{A}, \mathbf{B}]$ denotes concatenation.

A.2 Exact learning dynamics with prior knowledge

A.2.1 Balanced condition

Definition A.2.1 (Definition of λ-*balanced* property ([243], [185]))**.** The weights $\mathbf{W}_1, \mathbf{W}_2$ are λ-*balanced* if and only if there exists a **Balanced Coefficient** $\lambda \in \mathbb{R}$ such that

$$(\mathbf{W}_1, \mathbf{W}_2) = \mathbf{W}_2^T \mathbf{W}_2 - \mathbf{W}_1 \mathbf{W}_1^T = \lambda \mathbf{I} \tag{A.36}$$

where B is called the **Balanced Computation**.

For $\lambda = 0$ we have **Zero-Balanced** given as

A4 (*Zero-Balanced*). $\mathbf{W}_1(0)\mathbf{W}_1(0)^T = \mathbf{W}_2(0)^T \mathbf{W}_2(0)$.

Theorem A.2.2. ***Balanced Condition Persists Through Training*** *Suppose at initialisation*

$$\mathbf{W}_2(0)^T \mathbf{W}_2(0) - \mathbf{W}_1(0)\mathbf{W}_1(0)^T = \lambda \mathbf{I}. \tag{A.37}$$

Then for all $t \geq 0$

$$\mathbf{W}_2(t)^T \mathbf{W}_2(t) - \mathbf{W}_1(t)\mathbf{W}_1(t)^T = \lambda \mathbf{I}. \tag{A.38}$$

Proof. Consider:

$$\begin{aligned}
&\tau \frac{d}{dt}\left[\mathbf{W}_2(t)^T \mathbf{W}_2(t) - \mathbf{W}_1(t)\mathbf{W}_1(t)^T\right] = \\
&\left(\tau \frac{d}{dt}\mathbf{W}_2(t)\right)^T \mathbf{W}_2(t) + \mathbf{W}_2(t)^T \left(\tau \frac{d}{dt}\mathbf{W}_2(t)\right) \\
&\quad - \left(\tau \frac{d}{dt}\mathbf{W}_1(t)\right) \mathbf{W}_1(t)^T - \mathbf{W}_1(t) \left(\tau \frac{d}{dt}\mathbf{W}_1(t)\right)^T \\
&= \mathbf{W}_1(t)\left(\tilde{\mathbf{\Sigma}}^{yx} - \mathbf{W}_2(t)\mathbf{W}_1(t)\tilde{\mathbf{\Sigma}}^{xx}\right)^T \mathbf{W}_2(t) \\
&\quad + \mathbf{W}_2(t)^T \left(\tilde{\mathbf{\Sigma}}^{yx} - \mathbf{W}_2(t)\mathbf{W}_1(t)\tilde{\mathbf{\Sigma}}^{xx}\right) \mathbf{W}_1(t) \\
&\quad - \mathbf{W}_2(t)^T \left(\tilde{\mathbf{\Sigma}}^{yx} - \mathbf{W}_2(t)\mathbf{W}_1(t)\tilde{\mathbf{\Sigma}}^{xx}\right) \mathbf{W}_1(t) \\
&\quad - \mathbf{W}_1(t)\left(\tilde{\mathbf{\Sigma}}^{yx} - \mathbf{W}_2(t)\mathbf{W}_1(t)\tilde{\mathbf{\Sigma}}^{xx}\right) \mathbf{W}_2(t) \\
&= \mathbf{0}.
\end{aligned} \tag{A.39}$$

Note that $\mathbf{W}_2(t)^T \mathbf{W}_2(t) - \mathbf{W}_1(t)\mathbf{W}_1(t)^T$ is conserved for any initial value λ. □

A.2.2 Discussion assumptions

Whittened Inputs. Although the whitened input assumption is quite strong, it is commonly used in analytical work to obtain exact solutions, and much of the existing literature relies on these solutions [93, 39, 158]. While relaxing this assumption prevents the exact description of network dynamics, Kunin et al. [158] examine the implicit bias of the training trajectory without relying on whitened inputs. If the interpolating manifold is one-dimensional, the solution can be solved exactly in terms of λ. Their findings demonstrate a similar quantitative dependence on λ, governing the implicit bias transition between rich and lazy regimes. Furthermore, recent advancements, such as the ”decorrelated backpropagation” technique introduced by Dalm et al. [59] which whitens inputs during training, showing that optimising for whitened inputs can actually be done in practice and improve efficiency in real-world applications. Importantly, This study highlights that in certain real-world scenarios, whitening can provide a more optimal learning condition. This approaches emphasise the potential advantages of input whitening for downstream tasks, reinforcing the validity of our assumption.

Dimension. Previous works imposed specific dimensionality constraints. For example: Fukumizu [93] assumed equal input and output dimensions ($N_i = N_o$) while allowing a bottleneck in the hidden dimension ($N_h \leq N_i = N_o$). In out first paper [39], we extended these solutions to cases with unequal input and output dimensions ($N_i \neq N_o$) but restricted bottleneck networks ($N_h = min(N_i, N_o)$) and introduced an additional invertibility condition on the $\mathbf{B}$. In the follow up work we allow for unequal input and output $N_i \neq N_o$ and do not introduce an additional invertibility assumption. This flexibility expands the applicability of our framework to a wider range of architectures. The dimensional constraints can be further relaxed by increasing the hidden layer width N_h, as outlined in Theorem 3.4.5; however, this approach yields limited benefits. Increasing N_h does not substantially enhance the network’s expressivity, as the effective rank remains bounded. Another constraint of our approach is that it does not extend to deep network architectures. Previous research has explored how depth influences learning dynamics [243], however, assuming that the network aligns with the singular structure of the task. Although in our analysis, we restrict our focus to two-layer networks, we can achieve a precise understanding of the learning dynamics without the need for alignment. Recent studies, including those by Kunin et al. [158], have investigated the implicit biases present in deep networks. However, these findings have certain limitations, as they lack exact analytical solutions for the dynamics of deep networks. The assumptions about dimensionality are often tied to the analytical techniques employed.

Full rank In our first work, [39], we imposed a full-rank initialisation condition, defined as $\text{rank}(\mathbf{W}_2(0)\mathbf{W}_1(0)) = N_i = N_o$. However, this assumption is not necessary in our framework when lambda is non-zero.

Balancedness Assumption A significant departure from prior works is the relaxation of the balancedness assumption: Earlier studies, such as Fukumizu [93] and our first work [39] assumed strict zero-balancedness ($\mathbf{W}_1(0)\mathbf{W}_1(0)^T = \mathbf{W}_2(0)^T\mathbf{W}_2(0)$), which constrained the networks to the *rich* regime. Our approach generalises this to λ-balancedness, enabling exploration of the continuum between the *rich* and *lazy* regimes. While some efforts, such as Tarmoun et al. [272], have explored removing the zero-balanced constraint, their solutions were limited to unstable or mixed forms. In contrast, our methodology systematically studies different learning regimes by varying initialisation properties, particularly through the relative scale parameter in a stable and non-mixed form. This allows controlled transitions between regimes, advancing understanding of neural network behavior across the spectrum. Other studies, such as Kunin et al. [158] and Xu and Zheng [296] have also relaxed the balancedness assumption, though their analysis was restricted to single-output neuron settings. We emphasise the importance of this balanced quantity by rigorously proving that, in the averaging limit, standard network initialisations (e.g., LeCun initialisation [164], He initialisation) lead to λ-balanced behavior in the infinite-width limit. Specifically, the term $\mathbf{W}_1(0)\mathbf{W}_1(0)^T = \mathbf{W}_2(0)^T\mathbf{W}_2(0)$ converges to a scaled identity matrix. Furthermore, previous studies have demonstrated that the relative scaling of λ significantly impacts the learning regime in practical scenarios, highlighting the crucial role of dynamical studies of networks as a function of this parameter [158]. Several prior studies also rely on the assumption of small random initial weights [243]. This assumption posits that when the weight scale is small, alignment occurs rapidly, enabling a simplified analysis focused on the dynamics of the singular values. Under these conditions, the balanced condition remains nearly zero throughout training. However, many initialization schemes fall outside both the balanced regime and the small-weight assumption—scenarios that most existing theoretical frameworks struggle to address. These assumptions are thus primarily made for analytical tractability. Exploring dynamics beyond these constrained settings is a valuable direction for future research.

A.2.3 $\mathbf{QQ}^T$ diagonalisation

Lemma A.2.3. *If $\mathbf{F} = \mathbf{P}\boldsymbol{\Lambda}\mathbf{P}^T$ is symmetric and diagonalisable, then the matrix Riccati differential equation $\frac{d}{dt}(\mathbf{QQ}^T) = \mathbf{FQQ}^T + \mathbf{QQ}^T\mathbf{F} - (\mathbf{QQ}^T)^2$ with initialisation $\mathbf{QQ}^T(0) = \mathbf{Q}(0)\mathbf{Q}(0)^T$ has a unique solution for all $t \geq 0$, and the solution is given by*

$$\mathbf{QQ}^T(t) = e^{\mathbf{F}\frac{t}{\tau}}\mathbf{Q}(0)\left[\mathbf{I} + \mathbf{Q}(0)^T\mathbf{P}\left(\frac{e^{2\boldsymbol{\Lambda}\frac{t}{\tau}} - \mathbf{I}}{2\boldsymbol{\Lambda}}\right)\mathbf{P}^T\mathbf{Q}(0)\right]^{-1}\mathbf{Q}(0)^T e^{\mathbf{F}\frac{t}{\tau}}. \quad \text{(A.40)}$$

This is true even when there exists $\boldsymbol{\Lambda}_i = 0$.

Proof. First we show that there exists a unique solution to the initial value problem stated. This is true by Picard-Lindelöf theorem. Now we show that the provided solution satisfies the ODE. Let $\mathbf{L} = e^{\mathbf{F}\frac{t}{\tau}}\mathbf{Q}(0)$ and $\mathbf{C} = \mathbf{I} + \mathbf{Q}(0)^T\mathbf{P}\left(\frac{e^{2\boldsymbol{\Lambda}\frac{t}{\tau}} - \mathbf{I}}{2\boldsymbol{\Lambda}}\right)\mathbf{P}^T\mathbf{Q}(0)$ such that solution $\mathbf{QQ}^T(t) = \mathbf{LC}^{-1}\mathbf{L}^T$. The time derivative of $\mathbf{QQ}^T$ is then given by

$$\frac{d}{dt}(\mathbf{QQ}^T) = \frac{d}{dt}(\mathbf{L})\mathbf{C}^{-1}\mathbf{L}^T + \mathbf{L}\frac{d}{dt}(\mathbf{C}^{-1})\mathbf{L}^T + \mathbf{LC}^{-1}\frac{d}{dt}(\mathbf{L}^T). \quad \text{(A.41)}$$

Solving for these derivatives individually, we find

$$\frac{d}{dt}(\mathbf{L}) = \frac{d}{dt}e^{\mathbf{F}\frac{t}{\tau}}\mathbf{Q}(0) = \mathbf{F}e^{\mathbf{F}\frac{t}{\tau}}\mathbf{Q}(0) = \mathbf{FL}, \quad \text{(A.42)}$$

$$\frac{d}{dt}(\mathbf{C}^{-1}) = -\mathbf{C}^{-1}\frac{d}{dt}(\mathbf{C})\mathbf{C}^{-1} = -\mathbf{C}^{-1}\mathbf{Q}(0)^T\mathbf{P}\frac{d}{dt}\left(\frac{e^{2\mathbf{\Lambda}\frac{t}{\tau}}-\mathbf{I}}{2\mathbf{\Lambda}}\right)\mathbf{P}^T\mathbf{Q}(0)\mathbf{C}^{-1}. \tag{A.43}$$

We consider the derivative of the fraction serpately,

$$\frac{d}{dt}\left(\frac{e^{2\mathbf{\Lambda}\frac{t}{\tau}}-\mathbf{I}}{2\mathbf{\Lambda}}\right) = e^{2\mathbf{\Lambda}\frac{t}{\tau}}, \tag{A.44}$$

this is true even in the limit as $\lambda_i \to 0$. Plugging these derivatives back in we see that the solution satisfies the ODE. Lastly, let $t = 0$, we see that the the solution satisfies the initial conditions. □

A.2.4 F diagonalisation

Lemma A.2.4. *The eigendecomposition of* $\mathbf{F} = \mathbf{P}\mathbf{\Lambda}\mathbf{P}^T$ *where*

$$\mathbf{P} = \frac{1}{\sqrt{2}}\begin{pmatrix} \tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \tilde{\mathbf{V}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \sqrt{2}\tilde{\mathbf{V}}_\perp \\ \tilde{\mathbf{U}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}}) & -\tilde{\mathbf{U}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \sqrt{2}\tilde{\mathbf{U}}_\perp \end{pmatrix}, \quad \mathbf{\Lambda} = \begin{pmatrix} \tilde{\mathbf{S}}_\lambda & 0 & 0 \\ 0 & -\tilde{\mathbf{S}}_\lambda & 0 \\ 0 & 0 & \boldsymbol{\lambda}_\perp \end{pmatrix} \tag{A.45}$$

and the matrices $\tilde{\mathbf{S}}_\lambda$*,* $\boldsymbol{\lambda}_\perp$*,* $\tilde{\mathbf{H}}$*, and* $\tilde{\mathbf{G}}$ *are the diagonal matrices defined as:*

$$\tilde{\mathbf{S}}_\lambda = \sqrt{\tilde{\mathbf{S}}^2 + \frac{\lambda^2}{4}\mathbf{I}},\ \boldsymbol{\lambda}_\perp = \mathrm{sgn}(N_o - N_i)\frac{\lambda}{2}\mathbf{I},\ \tilde{\mathbf{H}} = \mathrm{sgn}(\lambda)\sqrt{\frac{\tilde{\mathbf{S}_\lambda} - \tilde{\mathbf{S}}}{\tilde{\mathbf{S}_\lambda} + \tilde{\mathbf{S}}}},\ \tilde{\mathbf{G}} = \frac{1}{\sqrt{\mathbf{I}+\tilde{\mathbf{H}}^2}}. \tag{A.46}$$

Beyond the invertibility of F, we need to understand the relationship between F and $Q(0)$. To do this the following lemma relates the structure between the SVD of the model with the SVD structure of the individual parameters.

Proof. We leave for the reader by computing

$$\mathbf{F} = \mathbf{P}\mathbf{\Lambda}\mathbf{P}^T. \tag{A.47}$$

□

A.2.5 Solution unequal-input-output

Theorem A.2.5. *Under the assumptions of whitened inputs (Assumption 1), lambda-balanced weights (Assumption 2), no bottleneck (Assumption 3), the temporal dynamics of* $\mathbf{QQ}^T$ *are*

$$\mathbf{QQ}^T(t) = \begin{pmatrix} \mathbf{Z}_1\mathbf{A}^{-1}\mathbf{Z}_1^T & \mathbf{Z}_1\mathbf{A}^{-1}\mathbf{Z}_2^T \\ \mathbf{Z}_2\mathbf{A}^{-1}\mathbf{Z}_1^T & \mathbf{Z}_2\mathbf{A}^{-1}\mathbf{Z}_2^T \end{pmatrix}, \tag{A.48}$$

where the variables $\mathbf{Z}_1 \in \mathbb{R}^{N_i \times N_h}$, $\mathbf{Z}_2 \in \mathbb{R}^{N_o \times N_h}$, *and* $\mathbf{A} \in \mathbb{R}^{N_h \times N_h}$ *are defined as*

$$\mathbf{Z}_1(t) = \frac{1}{2}\tilde{\mathbf{V}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^T - \frac{1}{2}\tilde{\mathbf{V}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T + \tilde{\mathbf{V}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T, \tag{A.49}$$

$$\mathbf{Z}_2(t) = \frac{1}{2}\tilde{\mathbf{U}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^T + \frac{1}{2}\tilde{\mathbf{U}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T + \tilde{\mathbf{U}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0), \tag{A.50}$$

$$\begin{aligned}\mathbf{A}(t) =& \mathbf{I} + \mathbf{B}\left(\frac{e^{2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} - \mathbf{I}}{4\tilde{\mathbf{S}}_\lambda}\right)\mathbf{B}^T - \mathbf{C}\left(\frac{e^{-2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} - \mathbf{I}}{4\tilde{\mathbf{S}}_\lambda}\right)\mathbf{C}^T + \mathbf{W}_2(0)^T\tilde{\mathbf{U}}_\perp\left(\frac{e^{\lambda_\perp \frac{t}{\tau}} - \mathbf{I}}{\lambda_\perp}\right)\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0) \\ &+ \mathbf{W}_1(0)\tilde{\mathbf{V}}_\perp\left(\frac{e^{\boldsymbol{\lambda}_\perp \frac{t}{\tau}} - \mathbf{I}}{\boldsymbol{\lambda}_\perp}\right)\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T.\end{aligned} \tag{A.51}$$

Proof. We start and use the diagonalisation of $\mathbf{F}$ to rewrite the matrix exponential of $\mathbf{F}$ and $\mathbf{F}$. Note that $\mathbf{P}^T\mathbf{P} = \mathbf{PP}^T = \mathbf{I}$ and therefore $\mathbf{P}^T = \mathbf{P}^{-1}$.

$$\begin{aligned} e^{\mathbf{F}\frac{t}{\tau}} &= \mathbf{P}e^{\mathbf{\Gamma}}\mathbf{P}^\intercal \\ &= \frac{1}{\sqrt{2}}\begin{bmatrix} \tilde{\mathbf{V}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \tilde{\mathbf{V}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \sqrt{2}\mathbf{V}_\perp \\ \tilde{\mathbf{U}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & -\tilde{\mathbf{U}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \sqrt{2}\mathbf{U}_\perp \end{bmatrix}\begin{bmatrix} e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} & 0 & 0 \\ 0 & e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} & 0 \\ 0 & 0 & e^{\boldsymbol{\lambda}_\perp \frac{t}{\tau}} \end{bmatrix} \\ &\times \frac{1}{\sqrt{2}}\begin{bmatrix} \tilde{\mathbf{V}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \tilde{\mathbf{V}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \sqrt{2}\mathbf{V}_\perp \\ \tilde{\mathbf{U}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & -\tilde{\mathbf{U}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \sqrt{2}\mathbf{U}_\perp \end{bmatrix}^T \\ &= \frac{1}{\sqrt{2}}\begin{bmatrix} \tilde{\mathbf{V}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \tilde{\mathbf{V}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}}) \\ \tilde{\mathbf{U}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & -\tilde{\mathbf{U}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}}) \end{bmatrix}\begin{bmatrix} e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} & 0 \\ 0 & e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} \end{bmatrix} \\ &\times \frac{1}{\sqrt{2}}\begin{bmatrix} \tilde{\mathbf{V}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \tilde{\mathbf{V}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}}) \\ \tilde{\mathbf{U}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & -\tilde{\mathbf{U}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}}) \end{bmatrix}^T \end{aligned} \tag{A.52}$$

$$+2\frac{1}{\sqrt{2}}\begin{bmatrix}\tilde{\mathbf{V}}_\perp\\ \tilde{\mathbf{U}}_\perp\end{bmatrix}e^{\boldsymbol{\lambda}_\perp\frac{t}{\tau}}\frac{1}{\sqrt{2}}\begin{bmatrix}\tilde{\mathbf{V}}_\perp\\ \tilde{\mathbf{U}}_\perp\end{bmatrix}^T$$

$$=\mathbf{O}e^{\boldsymbol{\Lambda}\frac{t}{\tau}}\mathbf{O}+2\mathbf{M}e^{\boldsymbol{\lambda}_\perp\frac{t}{\tau}}\mathbf{M}^T. \tag{A.53}$$

$$\mathbf{F}=\mathbf{O}\Lambda\mathbf{O}^T+2\mathbf{M}\boldsymbol{\lambda}_\perp\mathbf{M}^T. \tag{A.54}$$

$$e^{\mathbf{F}\frac{t}{\tau}}\mathbf{F}^{-1}e^{\mathbf{F}\frac{t}{\tau}}-\mathbf{F}^{-1}= \tag{A.55}$$

$$\mathbf{O}e^{\boldsymbol{\Lambda}\frac{t}{\tau}}\mathbf{O}^T\mathbf{O}\boldsymbol{\Lambda}^{-1}\mathbf{O}^T\mathbf{O}e^{\boldsymbol{\Lambda}\frac{t}{\tau}}\mathbf{O}^T-\mathbf{O}\boldsymbol{\Lambda}^{-1}\mathbf{O}^T+\mathbf{M}(e^{\boldsymbol{\lambda}_\perp\frac{t}{\tau}}-\mathbf{I})(\boldsymbol{\lambda}_\perp)^{-1}\mathbf{M}^T.$$

Where $\mathbf{M}=\frac{1}{\sqrt{2}}\begin{bmatrix}\tilde{\mathbf{V}}_\perp\\ \tilde{\mathbf{U}}_\perp\end{bmatrix}^T$. Placing these expressions into Eq. A.40 gives

$$\begin{aligned}\mathbf{Q}\mathbf{Q}^T(t)=&\left[\mathbf{O}e^{\boldsymbol{\Lambda}\frac{t}{\tau}}\mathbf{O}^T+2\mathbf{M}e^{\boldsymbol{\lambda}_\perp\frac{t}{\tau}}\mathbf{M}^T\right]\mathbf{Q}(0)\\ &\left[\mathbf{I}+\frac{1}{2}\mathbf{Q}(0)^T\left(\mathbf{O}\left(e^{2\boldsymbol{\Lambda}\frac{t}{\tau}}-\mathbf{I}\right)\boldsymbol{\Lambda}^{-1}\mathbf{O}^T+\mathbf{M}(e^{\boldsymbol{\lambda}_\perp\frac{t}{\tau}}-\mathbf{I})\boldsymbol{\lambda}_\perp^{-1}\mathbf{M}^T\right)\mathbf{Q}(0)\right]^{-1}\\ &\mathbf{Q}(0)^T\left[\mathbf{O}e^{\boldsymbol{\Lambda}\frac{t}{\tau}}\mathbf{O}^T+2\mathbf{M}e^{\boldsymbol{\lambda}_\perp\frac{t}{\tau}}\mathbf{M}^T\right]^T.\end{aligned} \tag{A.56}$$

$$\begin{aligned}\boldsymbol{O}^T\boldsymbol{Q}(0)&=\frac{1}{\sqrt{2}}\begin{pmatrix}\tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \tilde{\mathbf{V}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})\\ \tilde{\mathbf{U}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}}) & -\tilde{\mathbf{U}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})\end{pmatrix}^T\begin{pmatrix}\mathbf{W}_1^T(0)\\ \mathbf{W}_2(0)\end{pmatrix}\\ &=\frac{1}{\sqrt{2}}\begin{pmatrix}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})\tilde{\mathbf{V}}^T\mathbf{W}_1^T(0)+(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})\tilde{\mathbf{U}}^T\mathbf{W}_2(0)\\ (\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})\tilde{\mathbf{V}}^T\mathbf{W}_1^T(0)-(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})\tilde{\mathbf{U}}^T\mathbf{W}_2(0)\end{pmatrix}\\ &=\frac{1}{\sqrt{2}}\begin{pmatrix}\mathbf{B}^T\\ -\mathbf{C}^T\end{pmatrix},\end{aligned} \tag{A.57}$$

where

$$\mathbf{B} = \mathbf{W}_2(0)^T\tilde{\mathbf{U}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}}) + \mathbf{W}_1(0)\tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}}) \in \mathbb{R}^{N_h\times N_h}, \tag{A.58}$$

$$\mathbf{C} = \mathbf{W}_2(0)^T\tilde{\mathbf{U}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}}) - \mathbf{W}_1(0)\tilde{\mathbf{V}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}}) \in \mathbb{R}^{N_h\times N_h}. \tag{A.59}$$

$$\begin{aligned}\boldsymbol{O}e^{\boldsymbol{\Lambda}t/\tau} &= \frac{1}{\sqrt{2}}\begin{pmatrix}\tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \tilde{\mathbf{V}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})\\ \tilde{\mathbf{U}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}}) & -\tilde{\mathbf{U}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})\end{pmatrix}\begin{pmatrix}e^{\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}} & 0\\ 0 & e^{-\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\end{pmatrix}\\ &= \frac{1}{\sqrt{2}}\begin{pmatrix}\tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}} & \tilde{\mathbf{V}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\\ \tilde{\mathbf{U}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}} & -\tilde{\mathbf{U}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\end{pmatrix}.\end{aligned} \tag{A.60}$$

$$\begin{aligned}\boldsymbol{O}e^{\boldsymbol{\Lambda}t/\tau}\boldsymbol{O}^T\boldsymbol{Q}(0) &= \frac{1}{2}\begin{pmatrix}\tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}} & \tilde{\mathbf{V}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\\ \tilde{\mathbf{U}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}} & -\tilde{\mathbf{U}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\end{pmatrix}\begin{pmatrix}\mathbf{B}^T\\ -\mathbf{C}^T\end{pmatrix}\\ &= \frac{1}{2}\begin{pmatrix}\tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{B}^T - \tilde{\mathbf{V}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{C}^T\\ \tilde{\mathbf{U}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{B}^T + \tilde{\mathbf{U}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{C}^T\end{pmatrix}.\end{aligned} \tag{A.61}$$

$$\begin{aligned}2\mathbf{M}e^{\boldsymbol{\lambda}_\perp\frac{t}{\tau}}\mathbf{M}^T\mathbf{Q}(0) &= 2\frac{1}{\sqrt{2}}\begin{bmatrix}\tilde{\mathbf{V}}_\perp\\ \tilde{\mathbf{U}}_\perp\end{bmatrix}\begin{bmatrix}e^{\boldsymbol{\lambda}_\perp\frac{t}{\tau}} & 0\\ 0 & e^{\boldsymbol{\lambda}_\perp\frac{t}{\tau}}\end{bmatrix}\frac{1}{\sqrt{2}}\begin{bmatrix}\tilde{\mathbf{V}}_\perp\\ \tilde{\mathbf{U}}_\perp\end{bmatrix}^T\begin{bmatrix}\mathbf{W}_1(0)^T\\ \mathbf{W}_2(0)\end{bmatrix}\\ &= \begin{bmatrix}\tilde{\mathbf{V}}_\perp e^{\boldsymbol{\lambda}_\perp\frac{t}{\tau}}\tilde{\mathbf{V}}_\perp^T & 0\\ 0 & \tilde{\mathbf{U}}_\perp e^{\boldsymbol{\lambda}_\perp\frac{t}{\tau}}\tilde{\mathbf{U}}_\perp^T\end{bmatrix}\begin{bmatrix}\mathbf{W}_1(0)^T\\ \mathbf{W}_2(0)\end{bmatrix}\\ &= \begin{bmatrix}\tilde{\mathbf{V}}_\perp e^{\boldsymbol{\lambda}_\perp\frac{t}{\tau}}\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T\\ \tilde{\mathbf{U}}_\perp e^{\boldsymbol{\lambda}_\perp\frac{t}{\tau}}\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)\end{bmatrix}.\end{aligned} \tag{A.62}$$

Putting it together we get the expressions for $\mathbf{Z}_1(t)$ and $\mathbf{Z}_2(t)$

$$\left[\mathbf{O}e^{\boldsymbol{\Lambda}\frac{t}{\tau}}\mathbf{O}^T + 2\mathbf{M}e^{\boldsymbol{\lambda}_\perp\frac{t}{\tau}}\mathbf{M}^T\right]\mathbf{Q}(0) = \tag{A.63}$$

$$= \frac{1}{2}\begin{pmatrix}\tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{B}^T - \tilde{\mathbf{V}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{C}^T\\ \tilde{\mathbf{U}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{B}^T + \tilde{\mathbf{U}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{C}^T\end{pmatrix} + \begin{bmatrix}\tilde{\mathbf{V}}_\perp e^{\boldsymbol{\lambda}_\perp\frac{t}{\tau}}\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T\\ \tilde{\mathbf{U}}_\perp e^{\boldsymbol{\lambda}_\perp\frac{t}{\tau}}\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)\end{bmatrix}.$$

$$\mathbf{Z}_1(t) = \frac{1}{2}\tilde{\mathbf{V}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^T - \frac{1}{2}\tilde{\mathbf{V}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T + \tilde{\mathbf{V}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\tilde{\mathbf{V}}_\perp^T \mathbf{W}_1(0)^T. \tag{A.64}$$

$$\mathbf{Z}_2(t) = \frac{1}{2}\tilde{\mathbf{U}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^T + \frac{1}{2}\tilde{\mathbf{U}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T + \tilde{\mathbf{U}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\tilde{\mathbf{U}}_\perp^T \mathbf{W}_2(0). \tag{A.65}$$

We now compute the terms inside the inverse

$$\begin{aligned}
&\mathbf{Q}(0)^T\mathbf{M}(e^{\lambda_\perp \frac{t}{\tau}})\lambda_\perp^{-1}\mathbf{M}^T\mathbf{Q}(0) = \\
&\begin{bmatrix}\mathbf{W}_1(0) & \mathbf{W}_2(0)^T\end{bmatrix} \frac{1}{\sqrt{2}} \begin{bmatrix}\tilde{\mathbf{V}}_\perp \\ \tilde{\mathbf{U}}_\perp\end{bmatrix} \begin{bmatrix} e^{\lambda_\perp \frac{t}{\tau}} & 0 \\ 0 & e^{\lambda_\perp \frac{t}{\tau}}\end{bmatrix} \begin{bmatrix}\lambda_\perp & 0 \\ 0 & \lambda_\perp\end{bmatrix}^{-1} \frac{1}{\sqrt{2}} \begin{bmatrix}\tilde{\mathbf{V}}_\perp \\ \tilde{\mathbf{U}}_\perp\end{bmatrix}^T \begin{bmatrix}\mathbf{W}_1(0)^T \\ \mathbf{W}_2(0)\end{bmatrix} \\
&\begin{bmatrix}\mathbf{W}_1(0) & \mathbf{W}_2(0)^T\end{bmatrix} \begin{bmatrix} e^{\lambda_\perp \frac{t}{\tau}}\lambda_\perp^{-1}\tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T \\ e^{\lambda_\perp \frac{t}{\tau}}\lambda_\perp^{-1}\tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)\end{bmatrix} \\
&= \left[\left(\mathbf{W}_1(0)\tilde{\mathbf{V}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\lambda_\perp^{-1}\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T + \mathbf{W}_2(0)^T\tilde{\mathbf{U}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\lambda_\perp^{-1}\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)\right)\right].
\end{aligned} \tag{A.66}$$

$$\begin{aligned}
&\mathbf{Q}(0)^T\mathbf{M}\lambda_\perp^{-1}\mathbf{M}^T\mathbf{Q}(0) = \\
&2\begin{bmatrix}\mathbf{W}_1(0) & \mathbf{W}_2(0)^T\end{bmatrix} \frac{1}{\sqrt{2}} \begin{bmatrix}\tilde{\mathbf{V}}_\perp \\ \tilde{\mathbf{U}}_\perp\end{bmatrix} \begin{bmatrix}\lambda_\perp & 0 \\ 0 & \lambda_\perp\end{bmatrix}^{-1} \frac{1}{\sqrt{2}} \begin{bmatrix}\tilde{\mathbf{V}}_\perp \\ \tilde{\mathbf{U}}_\perp\end{bmatrix}^T \begin{bmatrix}\mathbf{W}_1(0)^T \\ \mathbf{W}_2(0)\end{bmatrix} \\
&= \begin{bmatrix}\mathbf{W}_1(0) & \mathbf{W}_2(0)^T\end{bmatrix} \begin{bmatrix}\tilde{\mathbf{V}}_\perp \\ \tilde{\mathbf{U}}_\perp\end{bmatrix} \begin{bmatrix}\lambda_\perp^{-1}\tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T \\ \lambda_\perp^{-1}\tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)\end{bmatrix} \\
&= \left[\mathbf{W}_1(0)\tilde{\mathbf{V}}_\perp\lambda_\perp^{-1}\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T + \mathbf{W}_2(0)^T\tilde{\mathbf{U}}_\perp\lambda_\perp^{-1}\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)\right].
\end{aligned} \tag{A.67}$$

Now

$$\frac{1}{2}\mathbf{Q}(0)^T\mathbf{O}\left(e^{2\mathbf{\Lambda}\frac{t}{\tau}} - \mathbf{I}\right)\mathbf{\Lambda}^{-1}\mathbf{O}^T = \quad \frac{1}{4}[\mathbf{B} - \mathbf{C}]\left(e^{\mathbf{\Lambda}\frac{t}{\tau}} - \mathbf{I}\right)\mathbf{\Lambda}^{-1}\begin{pmatrix}\mathbf{B}^T \\ -\mathbf{C}^T\end{pmatrix} \tag{A.68}$$

$$= \frac{1}{4}\left(\mathbf{B}\left(e^{2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} - \mathbf{I}\right)(\tilde{\mathbf{S}}_\lambda)^{-1}\mathbf{B}^T - \mathbf{C}\left(e^{-2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} - \mathbf{I}\right)(\tilde{\mathbf{S}}_\lambda)^{-1}\mathbf{C}^T\right). \tag{A.69}$$

Putting it all together

$$\begin{aligned}\mathbf{A}(t) =& \mathbf{I} + \mathbf{B}\left(\frac{e^{2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} - \mathbf{I}}{4\tilde{\mathbf{S}}_\lambda}\right)\mathbf{B}^T - \mathbf{C}\left(\frac{e^{-2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} - \mathbf{I}}{4\tilde{\mathbf{S}}_\lambda}\right)\mathbf{C}^T \\ &+ \mathbf{W}_2(0)^T\tilde{\mathbf{U}}_\perp\left(\frac{e^{\lambda_\perp \frac{t}{\tau}} - \mathbf{I}}{\lambda_\perp}\right)\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0) + \mathbf{W}_1(0)\tilde{\mathbf{V}}_\perp\left(\frac{e^{\lambda_\perp \frac{t}{\tau}} - \mathbf{I}}{\lambda_\perp}\right)\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T.\end{aligned} \tag{A.70}$$

So, final form:

$$\begin{aligned}&\mathbf{Q}\mathbf{Q}^T(t) = \\ &\left[\begin{pmatrix}\frac{1}{2}\tilde{\mathbf{V}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^T - \frac{1}{2}\tilde{\mathbf{V}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T + \tilde{\mathbf{V}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T \\ \frac{1}{2}\tilde{\mathbf{U}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^T + \frac{1}{2}\tilde{\mathbf{U}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T + \tilde{\mathbf{U}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)\end{pmatrix}\right] \\ &\left[\mathbf{I} + \frac{1}{4}\left(\mathbf{B}\left(\frac{e^{2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} - \mathbf{I}}{\tilde{\mathbf{S}}_\lambda}\right)\mathbf{B}^T - \mathbf{C}\left(\frac{e^{-2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} - \mathbf{I}}{\tilde{\mathbf{S}}_\lambda}\right)\mathbf{C}^T\right)\right. \\ &\left. + \mathbf{W}_2(0)^T\tilde{\mathbf{U}}_\perp\left(\frac{e^{\lambda_\perp \frac{t}{\tau}} - \mathbf{I}}{\lambda_\perp}\right)\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0) + \mathbf{W}_1(0)\tilde{\mathbf{V}}_\perp\left(\frac{e^{\lambda_\perp \frac{t}{\tau}} - \mathbf{I}}{\lambda_\perp}\right)\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T\right]^{-1} \\ &\left[\begin{pmatrix}\frac{1}{2}\tilde{\mathbf{V}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^T - \frac{1}{2}\tilde{\mathbf{V}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T + \tilde{\mathbf{V}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T \\ \frac{1}{2}\tilde{\mathbf{U}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^T + \frac{1}{2}\tilde{\mathbf{U}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T + \tilde{\mathbf{U}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)\end{pmatrix}\right]^T.\end{aligned} \tag{A.71}$$

□

A.2.6 Stable solution unequal-input-output

Theorem A.2.6. *Given the assumptions of Theorem 3.3.3 further assuming that* $\mathbf{B}$ *is invertible and defining* $e^{\lambda_\perp \frac{t}{\tau}} = \operatorname{sgn}(N_o - N_i)\frac{\lambda}{2}$, *the temporal evolution of* $\mathbf{Q}\mathbf{Q}^T$ *is described as follows:*

$$
\begin{aligned}
\mathbf{Q}\mathbf{Q}^T(t) = \mathbf{Z}\Big[& e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^{-1}\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} \\
&+ \left(\frac{\mathbf{I}-e^{-2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}}{4\tilde{\mathbf{S}}_\lambda}\right) - e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^{-1}\mathbf{C}\left(\frac{e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}-\mathbf{I}}{4\tilde{\mathbf{S}}_\lambda}\right)\mathbf{C}^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} \\
&- e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^{-1}\mathbf{W}_2(0)^T\tilde{\mathbf{U}}_\perp\lambda_\perp^{-1}\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} \\
&e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}e^{\frac{\lambda_\perp}{2}\frac{t}{\tau}}\mathbf{B}^{-1}\mathbf{W}_2(0)^T\tilde{\mathbf{U}}_\perp\lambda_\perp^{-1}\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} \\
&+ e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}e^{\frac{\lambda}{2}\frac{t}{\tau}}\mathbf{B}^{-1}\mathbf{W}_1(0)\tilde{\mathbf{V}}_\perp\lambda_\perp^{-1}\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} \\
&- e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^{-1}\mathbf{W}_1(0)\tilde{\mathbf{V}}_\perp\lambda_\perp^{-1}\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\Big]^{-1}\mathbf{Z}^T,
\end{aligned}
\tag{A.72}
$$

where

$$
\mathbf{Z} = \begin{pmatrix}
\frac{1}{2}\tilde{\mathbf{V}}\left[(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})-(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\right]+\tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)\mathbf{B}^{-T}e^{\lambda_\perp \frac{t}{\tau}}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} \\
\frac{1}{2}\tilde{\mathbf{U}}\left[(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})+(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\right]+\tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)^T\mathbf{B}^{-T}e^{\lambda_\perp \frac{t}{\tau}}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}.
\end{pmatrix}
\tag{A.73}
$$

Proof. We start from

$$
\begin{aligned}
&\mathbf{Q}\mathbf{Q}^T(t) = \\
&\left[\begin{pmatrix}
\frac{1}{2}\tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^T-\frac{1}{2}\tilde{\mathbf{V}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T+\tilde{\mathbf{V}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T \\
\frac{1}{2}\tilde{\mathbf{U}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^T+\frac{1}{2}\tilde{\mathbf{U}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T+\tilde{\mathbf{U}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)
\end{pmatrix}\right] \\
&\left[\mathbf{I}+\frac{1}{4}\left(\mathbf{B}\left(\frac{e^{2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}-\mathbf{I}}{\tilde{\mathbf{S}}_\lambda}\right)\mathbf{B}^T-\mathbf{C}\left(\frac{e^{-2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}-\mathbf{I}}{\tilde{\mathbf{S}}_\lambda}\right)\mathbf{C}^T\right)\right. \\
&\left.+\mathbf{W}_2(0)^T\tilde{\mathbf{U}}_\perp\left(\frac{e^{\lambda_\perp \frac{t}{\tau}}-\mathbf{I}}{\lambda_\perp}\right)\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)+\mathbf{W}_1(0)\tilde{\mathbf{V}}_\perp\left(\frac{e^{\lambda_\perp \frac{t}{\tau}}-\mathbf{I}}{\lambda_\perp}\right)\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T\right]^{-1} \\
&\left[\begin{pmatrix}
\frac{1}{2}\tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^T-\frac{1}{2}\tilde{\mathbf{V}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T+\tilde{\mathbf{V}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T \\
\frac{1}{2}\tilde{\mathbf{U}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^T+\frac{1}{2}\tilde{\mathbf{U}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T+\tilde{\mathbf{U}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)
\end{pmatrix}\right]^T.
\end{aligned}
$$

We extract $\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}$ from all terms as exemplified bellow

$$\mathbf{O}e^{\mathbf{\Lambda}t/\tau}\mathbf{O}^T\mathbf{Q}(0) = \frac{1}{2}\begin{pmatrix} \tilde{\mathbf{V}}\left[(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}}) - (\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\right] \\ \tilde{\mathbf{U}}\left[(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}}) + (\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\right] \end{pmatrix}\mathbf{B}^T e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}, \tag{A.74}$$

and rewrite the dynamics as

$$\mathbf{Q}\mathbf{Q}^T(t) = \tag{A.75}$$

$$\left[\begin{pmatrix} \frac{1}{2}\tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}}) - \frac{1}{2}\tilde{\mathbf{V}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} + \tilde{\mathbf{V}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} \\ \frac{1}{2}\tilde{\mathbf{U}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}}) + \frac{1}{2}\tilde{\mathbf{U}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} + \tilde{\mathbf{U}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} \end{pmatrix}\right]$$

$$\left[e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^{-1}\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} + \frac{1}{4}\left(\left(\frac{\mathbf{I}-e^{-2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}}{\tilde{\mathbf{S}}_\lambda}\right) - e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^{-1}\mathbf{C}\left(\frac{e^{-2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}-\mathbf{I}}{\tilde{\mathbf{S}}_\lambda}\right)\mathbf{C}^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\right)\right.$$

$$+ e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^{-1}\mathbf{W}_2(0)^T\tilde{\mathbf{U}}_\perp\left(\frac{e^{\lambda_\perp \frac{t}{\tau}}-\mathbf{I}}{\lambda_\perp}\right)\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}$$

$$\left. + e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^{-1}\mathbf{W}_1(0)\tilde{\mathbf{V}}_\perp\left(\frac{e^{\lambda_\perp \frac{t}{\tau}}-\mathbf{I}}{\lambda_\perp}\right)\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\right]^{-1}$$

$$\left[\begin{pmatrix} \frac{1}{2}\tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}}) - \frac{1}{2}\tilde{\mathbf{V}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} + \tilde{\mathbf{V}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} \\ \frac{1}{2}\tilde{\mathbf{U}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}}) + \frac{1}{2}\tilde{\mathbf{U}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} + \tilde{\mathbf{U}}_\perp e^{\lambda_\perp \frac{t}{\tau}}\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} \end{pmatrix}\right]^T$$

$$= \begin{pmatrix} \frac{1}{2}\tilde{\mathbf{V}}\left[(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}}) - (\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\right] + \tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)\mathbf{B}^{-T}e^{\lambda_\perp \frac{t}{\tau}}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} \\ \frac{1}{2}\tilde{\mathbf{U}}\left[(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}}) + (\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\right] + \tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)^T\mathbf{B}^{-T}e^{\lambda_\perp \frac{t}{\tau}}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} \end{pmatrix}$$

$$\left[e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^{-1}\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\right.$$

$$+ \left(\frac{\mathbf{I}-e^{-2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}}{4\tilde{\mathbf{S}}_\lambda}\right) - e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^{-1}\mathbf{C}\left(\frac{e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}-\mathbf{I}}{4\tilde{\mathbf{S}}_\lambda}\right)\mathbf{C}^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}$$

$$- e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^{-1}\mathbf{W}_2(0)^T\tilde{\mathbf{U}}_\perp\lambda_\perp^{-1}\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}$$

$$e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}e^{\frac{\lambda_\perp}{2}\frac{t}{\tau}}\mathbf{B}^{-1}\mathbf{W}_2(0)^T\tilde{\mathbf{U}}_\perp\lambda_\perp^{-1}\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}$$

$$+ e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}e^{\frac{\lambda}{2}\frac{t}{\tau}}\mathbf{B}^{-1}\mathbf{W}_1(0)\tilde{\mathbf{V}}_\perp\lambda_\perp^{-1}\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}$$

$$\left. - e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{B}^{-1}\mathbf{W}_1(0)\tilde{\mathbf{V}}_\perp\lambda_\perp^{-1}\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\right]^{-1}$$

$$\begin{pmatrix} \tilde{\mathbf{V}}\left[(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}}) - (\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\right] + \tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T\mathbf{W}_1(0)\mathbf{B}^{-T}e^{\lambda_\perp \frac{t}{\tau}}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} \\ \tilde{\mathbf{U}}\left[(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}}) + (\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{C}^T\mathbf{B}^{-T}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\right] + \tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\mathbf{W}_2(0)^T\mathbf{B}^{-T}e^{\lambda_\perp \frac{t}{\tau}}e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} \end{pmatrix}^T.$$

where $e^{\lambda_\perp \frac{t}{\tau}} = \text{sgn}(N_o - N_i)\frac{\lambda}{2}$ is a scalar

□

A.2.6.1 Proof exact learning dynamics with prior knowledge unequal dimension

We start with the following equation

$$\begin{aligned}
\mathbf{QQ}^T(t) = &\underbrace{\left[\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T + 2\mathbf{M}e^{\lambda_\perp\frac{t}{\tau}}\mathbf{M}^T\right]\mathbf{Q}(0)}_{\mathbf{L}} \\
&\underbrace{\left[\mathbf{I} + \frac{1}{2}\mathbf{Q}(0)^T\left(\mathbf{O}\left(e^{2\mathbf{\Lambda}\frac{t}{\tau}} - \mathbf{I}\right)\mathbf{\Lambda}^{-1}\mathbf{O}^T + \mathbf{M}(e^{\lambda_\perp\frac{t}{\tau}} - \mathbf{I})\lambda_\perp^{-1}\mathbf{M}^T\right)\mathbf{Q}(0)\right]^{-1}}_{\mathbf{C}^{-1}} \\
&\underbrace{\mathbf{Q}(0)^T\left[\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T + 2\mathbf{M}e^{\lambda_\perp\frac{t}{\tau}}\mathbf{M}^T\right]}_{\mathbf{R}} \\
=&\mathbf{LC}^{-1}\mathbf{R}. \qquad \text{(A.76)}
\end{aligned}$$

Substituting our solution into the matrix Riccati equation then yields

$$\tau\frac{d}{dt}\mathbf{QQ}^T = \mathbf{FQQ}^T + \mathbf{QQ}^T\mathbf{F} - (\mathbf{QQ}^T)^2, \qquad \text{(A.77)}$$

$$\Rightarrow \tau\frac{d}{dt}\mathbf{LC}^{-1}\mathbf{R} \overset{?}{=} \mathbf{FLC}^{-1}\mathbf{R} + \mathbf{LC}^{-1}\mathbf{RF} - \mathbf{LC}^{-1}\mathbf{RLC}^{-1}\mathbf{R}. \qquad \text{(A.78)}$$

Using the chain rule $\partial(\mathbf{AB}) = (\partial\mathbf{A})\mathbf{B} + \mathbf{A}(\partial\mathbf{B})$ and the identities

$$\frac{d}{dt}(\mathbf{A}^{-1}) = \mathbf{A}^{-1}(\frac{d}{dt}\mathbf{A})\mathbf{A}^{-1} \qquad \text{and} \qquad \frac{d}{dt}(e^{t\mathbf{A}}) = \mathbf{A}e^{t\mathbf{A}} = e^{t\mathbf{A}}\mathbf{A}. \qquad \text{(A.79)}$$

Next, we note that

$$\begin{aligned}
&\mathbf{O}^T\mathbf{O} = \\
&\frac{1}{\sqrt{2}}\begin{pmatrix} \tilde{\mathbf{V}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \tilde{\mathbf{V}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}}) \\ \tilde{\mathbf{U}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & -\tilde{\mathbf{U}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}}) \end{pmatrix}^T \frac{1}{\sqrt{2}}\begin{pmatrix} \tilde{\mathbf{V}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \tilde{\mathbf{V}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}}) \\ \tilde{\mathbf{U}}(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}}) & -\tilde{\mathbf{U}}(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}}) \end{pmatrix} \\
&= \mathbf{I}. \qquad \text{(A.80)}
\end{aligned}$$

$$\begin{aligned}
\mathbf{O}^T\mathbf{M} &= \frac{1}{\sqrt{2}}\begin{bmatrix}\tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \tilde{\mathbf{V}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}}) \\ \tilde{\mathbf{U}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}}) & -\tilde{\mathbf{U}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})\end{bmatrix}\frac{1}{\sqrt{2}}\begin{bmatrix}\tilde{\mathbf{V}}_\perp \\ \tilde{\mathbf{U}}_\perp\end{bmatrix} \\
&= \frac{1}{2}\begin{bmatrix}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})^T\tilde{\mathbf{V}}^T\tilde{\mathbf{V}}_\perp + (\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})^T\tilde{\mathbf{U}}^T\tilde{\mathbf{U}}_\perp \\ (\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})^T\tilde{\mathbf{V}}^T\tilde{\mathbf{V}}_\perp - (\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})^T\tilde{\mathbf{U}}^T\tilde{\mathbf{U}}_\perp\end{bmatrix} \\
&= \mathbf{0}.
\end{aligned} \tag{A.81}$$

$$\begin{aligned}
\mathbf{M}^T\mathbf{O} &= \frac{1}{\sqrt{2}}\begin{bmatrix}\tilde{\mathbf{V}}_\perp^T & \tilde{\mathbf{U}}_\perp^T\end{bmatrix}\begin{pmatrix}\tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}}) & \tilde{\mathbf{V}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}}) \\ \tilde{\mathbf{U}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}}) & -\tilde{\mathbf{U}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})\end{pmatrix} \\
&= \frac{1}{2}\begin{bmatrix}\tilde{\mathbf{V}}_\perp^T\tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}}) + \tilde{\mathbf{U}}_\perp^T\tilde{\mathbf{U}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}}) \\ \tilde{\mathbf{V}}_\perp^T\tilde{\mathbf{V}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}}) - \tilde{\mathbf{U}}_\perp^T\tilde{\mathbf{U}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})\end{bmatrix} \\
&= \mathbf{0}.
\end{aligned} \tag{A.82}$$

We get

$$\begin{aligned}
\tau\frac{d}{dt}\mathbf{Q}\mathbf{Q}^T &= \tau\frac{d}{dt}\left(\mathbf{L}\mathbf{C}^{-1}\mathbf{R}\right) \\
&= \tau\left(\frac{d}{dt}\mathbf{L}\right)\mathbf{C}^{-1}\mathbf{R} + \tau\mathbf{L}\left(\frac{d}{dt}C^{-1}\mathbf{R}\right) \\
&= \tau\left(\frac{d}{dt}\mathbf{L}\right)\mathbf{C}^{-1}\mathbf{R} + \tau\mathbf{L}\mathbf{C}^{-1}\left(\frac{d}{dt}\mathbf{R}\right) + \tau\mathbf{L}\left(\frac{d}{dt}\mathbf{C}^{-1}\right)\mathbf{R},
\end{aligned} \tag{A.83}$$

with

$$\begin{aligned}
\tau\left(\frac{d}{dt}\mathbf{L}\right)\mathbf{C}^{-1}\mathbf{R} &= \tau\left(\mathbf{O}\frac{1}{\tau}\mathbf{\Lambda}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T + 2\mathbf{M}\frac{\lambda_\perp\mathbf{I}}{2\tau}e^{\lambda_\perp\frac{t}{\tau}}\mathbf{M}^T\right)\mathbf{Q}(0)\mathbf{C}^{-1}\mathbf{R} \\
&= \left(\mathbf{O}\mathbf{\Lambda}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T + \mathbf{M}\lambda_\perp\mathbf{I}e^{\lambda_\perp\frac{t}{\tau}}\mathbf{M}^T\right)\mathbf{Q}(0)\mathbf{C}^{-1}\mathbf{R} \\
&= (\mathbf{O}\lambda_\perp\mathbf{O}^T + 2\mathbf{M}\lambda_\perp\mathbf{M}^T)\left(\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T + 2\mathbf{M}e^{\lambda_\perp\frac{t}{\tau}}\mathbf{M}^T\right)\mathbf{Q}(0)\mathbf{C}^{-1}\mathbf{R} \\
&= \mathbf{F}\mathbf{L}\mathbf{C}^{-1}\mathbf{R},
\end{aligned} \tag{A.84}$$

$$\begin{aligned}
\tau\mathbf{L}\mathbf{C}^{-1}\left(\frac{d}{dt}\mathbf{R}\right) &= \tau\mathbf{L}\mathbf{C}^{-1}\mathbf{Q}(0)^T\left(\mathbf{O}\frac{1}{\tau}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{\Lambda}\mathbf{O}^T + 2\mathbf{M}e^{\lambda_\perp\frac{t}{\tau}}\frac{\lambda_\perp\mathbf{I}}{2\tau}\mathbf{M}^T\right) \\
&= \mathbf{L}\mathbf{C}^{-1}\mathbf{Q}(0)^T\left(\mathbf{O}\frac{1}{\tau}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{\Lambda}\mathbf{O}^T + 2\mathbf{M}e^{\lambda_\perp\frac{t}{\tau}}\frac{\lambda_\perp\mathbf{I}}{2\tau}\mathbf{M}^T\right) \\
&= \mathbf{L}\mathbf{C}^{-1}\mathbf{R}\mathbf{F}, \qquad\qquad (A.85)
\end{aligned}$$

and

$$\begin{aligned}
&\tau\mathbf{L}\left(\frac{d}{dt}\mathbf{C}^{-1}\right)\mathbf{R} = \\
&-\tau\mathbf{L}\mathbf{C}^{-1}\left(\frac{d}{dt}\mathbf{C}\right)\mathbf{C}^{-1}\mathbf{R} \\
&= -\mathbf{L}\mathbf{C}^{-1}\Bigg[\tau\frac{1}{2}\mathbf{Q}(0)^T\mathbf{O}2\frac{1}{\tau}e^{2\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{\Lambda}\mathbf{\Lambda}^{-1}\mathbf{O}^T\mathbf{Q}(0) \\
&\qquad\qquad + \tau\frac{1}{2}\mathbf{Q}(0)^T4\frac{1}{\tau}\mathbf{M}e^{\lambda_\perp\frac{t}{\tau}}\lambda_\perp(\lambda_\perp)^{-1}\mathbf{M}^T\mathbf{Q}(0)\Bigg]\mathbf{C}^{-1}\mathbf{R} \\
&= -\mathbf{L}\mathbf{C}^{-1}\left[\mathbf{Q}(0)^T\mathbf{O}e^{2\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T\mathbf{Q}(0) + 2\mathbf{Q}(0)^T\mathbf{M}e^{\lambda_\perp\frac{t}{\tau}}\mathbf{M}^T\mathbf{Q}(0)\right]\mathbf{C}^{-1}\mathbf{R} \\
&= -\mathbf{L}\mathbf{C}^{-1}\Bigg[\mathbf{Q}(0)^T\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T\mathbf{Q}(0) \\
&\qquad\qquad + 2\mathbf{Q}(0)^T\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\underbrace{\mathbf{O}^T\mathbf{M}}_{\mathbf{0}}e^{\lambda_\perp\frac{t}{\tau}}\mathbf{M}^T\mathbf{Q}(0) \\
&\qquad\qquad + 2\mathbf{Q}(0)^T\mathbf{M}e^{\lambda_\perp\frac{t}{\tau}}\underbrace{\mathbf{M}^T\mathbf{O}}_{\mathbf{0}}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T\mathbf{Q}(0) \\
&\qquad\qquad + 4\mathbf{Q}(0)^T\mathbf{M}e^{\lambda_\perp\frac{t}{\tau}}\mathbf{M}^T\mathbf{M}e^{\lambda_\perp\frac{t}{\tau}}\mathbf{M}^T\mathbf{Q}(0)\Bigg]\mathbf{C}^{-1}\mathbf{R} \\
&= -\mathbf{L}\mathbf{C}^{-1}\mathbf{R}\mathbf{L}\mathbf{C}^{-1}\mathbf{R}. \qquad\qquad (A.86)
\end{aligned}$$

Finally, substituting Eq. A.84, Eq. A.85 and Eq.A.86 into the left hand side of Eq. A.78 proves equality. □

A.2.7 Lambda equal zero case

In this setting, we impose a full-rank initialisation condition, defined as $\text{rank}(\mathbf{W}_2(0)\mathbf{W}_1(0)) = N_i = N_o$. The solution to the matrix Riccati equation as provided by Fukumizu [93] requires calculation of the inverse $\mathbf{F}^{-1}$ and the matrix exponential $e^{\mathbf{F}\frac{t}{\tau}}$. To this end, we diagonalise $\mathbf{F}$ by completing its basis by incorporating zero eigenvalues as illustrated below

$$
\begin{aligned}
\mathbf{F} &= \begin{bmatrix} 0 & \tilde{\mathbf{V}}\tilde{\mathbf{S}}\tilde{\mathbf{U}}^T \\ \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T & 0 \end{bmatrix} \\
&= \frac{1}{\sqrt{2}} \begin{bmatrix} \tilde{\mathbf{V}} & \tilde{\mathbf{V}} & \sqrt{2}\tilde{\mathbf{V}}_\perp \\ \tilde{\mathbf{U}} & -\tilde{\mathbf{U}} & \sqrt{2}\tilde{\mathbf{U}}_\perp \end{bmatrix} \begin{bmatrix} \tilde{\mathbf{S}} & 0 & 0 \\ 0 & -\tilde{\mathbf{S}} & 0 \\ 0 & 0 & 0 \end{bmatrix} \frac{1}{\sqrt{2}} \begin{bmatrix} \tilde{\mathbf{V}} & \tilde{\mathbf{V}} & \sqrt{2}\tilde{\mathbf{V}}_\perp \\ \tilde{\mathbf{U}} & -\tilde{\mathbf{U}} & \sqrt{2}\tilde{\mathbf{U}}_\perp \end{bmatrix}^T \\
&= \mathbf{P}\boldsymbol{\Gamma}\mathbf{P}^T. \qquad \text{(A.87)}
\end{aligned}
$$

Note that $\mathbf{P}^T\mathbf{P} = \mathbf{P}\mathbf{P}^T = \mathbf{I}$ and therefore $\mathbf{P}^T = \mathbf{P}^{-1}$. We then use the diagonalisation of $\mathbf{F}$ to rewrite the matrix exponential.

$$
\begin{aligned}
e^{\mathbf{F}\frac{t}{\tau}} &= \mathbf{P}e^{\boldsymbol{\Gamma}}\mathbf{P}^T \\
&= \frac{1}{\sqrt{2}} \begin{bmatrix} \tilde{\mathbf{V}} & \tilde{\mathbf{V}} & \sqrt{2}\mathbf{V}_\perp \\ \tilde{\mathbf{U}} & -\tilde{\mathbf{U}} & \sqrt{2}\mathbf{U}_\perp \end{bmatrix} \begin{bmatrix} e^{\tilde{\mathbf{S}}\frac{t}{\tau}} & 0 & 0 \\ 0 & e^{-\tilde{\mathbf{S}}\frac{t}{\tau}} & 0 \\ 0 & 0 & e^{0} \end{bmatrix} \frac{1}{\sqrt{2}} \begin{bmatrix} \tilde{\mathbf{V}} & \tilde{\mathbf{V}} & \sqrt{2}\mathbf{V}_\perp \\ \tilde{\mathbf{U}} & -\tilde{\mathbf{U}} & \sqrt{2}\mathbf{U}_\perp \end{bmatrix}^T \\
&= \frac{1}{2} \begin{bmatrix} \tilde{\mathbf{V}}e^{\tilde{\mathbf{S}}\frac{t}{\tau}}\tilde{\mathbf{V}}^T + \tilde{\mathbf{V}}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\tilde{\mathbf{V}}^T + 2\tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T & \tilde{\mathbf{V}}e^{\tilde{\mathbf{S}}\frac{t}{\tau}}\tilde{\mathbf{U}}^T - \tilde{\mathbf{V}}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\tilde{\mathbf{U}}^T + 2\tilde{\mathbf{V}}_\perp\tilde{\mathbf{U}}_\perp^T \\ \tilde{\mathbf{U}}e^{\tilde{\mathbf{S}}\frac{t}{\tau}}\tilde{\mathbf{V}}^T - \tilde{\mathbf{U}}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\tilde{\mathbf{V}}^T + 2\tilde{\mathbf{U}}_\perp\tilde{\mathbf{V}}_\perp^T & \tilde{\mathbf{U}}e^{\tilde{\mathbf{S}}\frac{t}{\tau}}\tilde{\mathbf{U}}^T - \tilde{\mathbf{U}}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\tilde{\mathbf{U}}^T + 2\tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T \end{bmatrix} \\
&= \frac{1}{\sqrt{2}} \begin{bmatrix} \tilde{\mathbf{V}} & \tilde{\mathbf{V}} \\ \tilde{\mathbf{U}} & -\tilde{\mathbf{U}} \end{bmatrix} \begin{bmatrix} e^{\tilde{\mathbf{S}}\frac{t}{\tau}} & 0 \\ 0 & e^{-\tilde{\mathbf{S}}\frac{t}{\tau}} \end{bmatrix} \frac{1}{\sqrt{2}} \begin{bmatrix} \tilde{\mathbf{V}} & \tilde{\mathbf{V}} \\ \tilde{\mathbf{U}} & -\tilde{\mathbf{U}} \end{bmatrix}^T + 2\frac{1}{\sqrt{2}} \begin{bmatrix} \tilde{\mathbf{V}}_\perp \\ \tilde{\mathbf{U}}_\perp \end{bmatrix} \frac{1}{\sqrt{2}} \begin{bmatrix} \tilde{\mathbf{V}}_\perp \\ \tilde{\mathbf{U}}_\perp \end{bmatrix}^T \\
&= \mathbf{O}e^{\boldsymbol{\Lambda}\frac{t}{\tau}}\mathbf{O} + 2\mathbf{M}\mathbf{M}^T. \qquad \text{(A.88)}
\end{aligned}
$$

As the inverse $\mathbf{F}^{-1} = \mathbf{P}\boldsymbol{\Gamma}^{-1}\mathbf{P}^T$ is not well defined for a $\boldsymbol{\Gamma}$ with zero eigenvalues. We study eigenvalues of value zero by analysing the limiting behaviour of $e^{\mathbf{F}\frac{t}{\tau}}\mathbf{F}^{-1}e^{\mathbf{F}\frac{t}{\tau}} - \mathbf{F}^{-1}$ for a single mode

$$
\lim_{\varepsilon\to 0}\left[e^{\frac{\varepsilon t}{\tau}}\frac{1}{\varepsilon}e^{\frac{\varepsilon t}{\tau}} - \frac{1}{\varepsilon}\right] = \lim_{\varepsilon\to 0}\left[\frac{e^{\frac{2\varepsilon t}{\tau}} - 1}{\varepsilon}\right] \qquad \text{(A.89)}
$$

$$
\xrightarrow{\text{L'Hospital}} \lim_{\varepsilon\to 0}\left[\frac{\frac{\partial}{\partial\varepsilon}\left(e^{\frac{2\varepsilon t}{\tau}} - 1\right)}{\frac{\partial}{\partial\varepsilon}\varepsilon}\right] = \lim_{\varepsilon\to 0} 2\frac{t}{\tau}e^{\frac{2\varepsilon t}{\tau}} = 2\frac{t}{\tau}. \qquad \text{(A.90)}
$$

which reveals the time dependent contribution of zero eigenvalues. Thus in the case where lambda tend to zero we have

$$e^{\mathbf{F}\frac{t}{\tau}}\mathbf{F}^{-1}e^{\mathbf{F}\frac{t}{\tau}} - \mathbf{F}^{-1} = \mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T\mathbf{O}\mathbf{\Lambda}^{-1}\mathbf{O}^T\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T - \mathbf{O}\mathbf{\Lambda}^{-1}\mathbf{O}^T + 4\frac{t}{\tau}\mathbf{M}\mathbf{M}^T. \tag{A.91}$$

We continue by placing these expressions into Eq. A.40 gives

$$\begin{aligned}
&\mathbf{Q}\mathbf{Q}^T(t) = \\
&\left[\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T + 2\mathbf{M}\mathbf{M}^T\right]\mathbf{Q}(0) \\
&\left[\mathbf{I} + \frac{1}{2}\mathbf{Q}(0)^T\left(\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T\mathbf{O}\mathbf{\Lambda}^{-1}\mathbf{O}^T\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T - \mathbf{O}\mathbf{\Lambda}^{-1}\mathbf{O}^T + 4\frac{t}{\tau}\mathbf{M}\mathbf{M}^T\right)\mathbf{Q}(0)\right]^{-1} \\
&\mathbf{Q}(0)^T\left[\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T + 2\mathbf{M}\mathbf{M}^T\right] \\
=&\left[\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T + 2\mathbf{M}\mathbf{M}^T\right]\mathbf{Q}(0) \\
&\left[\mathbf{I} + \frac{1}{2}\mathbf{Q}(0)^T\left(\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{\Lambda}^{-1}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T - \mathbf{O}\mathbf{\Lambda}^{-1}\mathbf{O}^T + 4\frac{t}{\tau}\mathbf{M}\mathbf{M}^T\right)\mathbf{Q}(0)\right]^{-1} \\
&\mathbf{Q}(0)^T\left[\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T + 2\mathbf{M}\mathbf{M}^T\right] \\
=&\left[\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T + 2\mathbf{M}\mathbf{M}^T\right]\mathbf{Q}(0) \\
&\left[\mathbf{I} + \frac{1}{2}\mathbf{Q}(0)^T\left(\mathbf{O}\left(e^{2\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{\Lambda}^{-1} - \mathbf{\Lambda}^{-1}\right)\mathbf{O}^T + 4\frac{t}{\tau}\mathbf{M}\mathbf{M}^T\right)\mathbf{Q}(0)\right]^{-1} \\
&\mathbf{Q}(0)^T\left[\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T + 2\mathbf{M}\mathbf{M}^T\right] \\
&= \left[\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T + 2\mathbf{M}\mathbf{M}^T\right]\mathbf{Q}(0) \\
&\left[\mathbf{I} + \frac{1}{2}\mathbf{Q}(0)^T\left(\mathbf{O}\left(e^{2\mathbf{\Lambda}\frac{t}{\tau}} - \mathbf{I}\right)\mathbf{\Lambda}^{-1}\mathbf{O}^T + 4\frac{t}{\tau}\mathbf{M}\mathbf{M}^T\right)\mathbf{Q}(0)\right]^{-1} \\
&\mathbf{Q}(0)^T\left[\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T + 2\mathbf{M}\mathbf{M}^T\right].
\end{aligned} \tag{A.92}$$

Then, matrix multiplication on the left side of the equation yields

$$\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}} = \frac{1}{\sqrt{2}}\begin{bmatrix}\tilde{\mathbf{V}} & \tilde{\mathbf{V}} \\ \tilde{\mathbf{U}} & -\tilde{\mathbf{U}}\end{bmatrix}\begin{bmatrix}e^{\tilde{\mathbf{S}}\frac{t}{\tau}} & 0 \\ 0 & e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix}\tilde{\mathbf{V}}e^{\tilde{\mathbf{S}}\frac{t}{\tau}} & \tilde{\mathbf{V}}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}} \\ \tilde{\mathbf{U}}e^{\tilde{\mathbf{S}}\frac{t}{\tau}} & -\tilde{\mathbf{U}}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\end{bmatrix} \tag{A.93}$$

and

$$\begin{aligned}
\mathbf{O}^T\mathbf{Q}(0) &= \frac{1}{\sqrt{2}}\begin{bmatrix}\tilde{\mathbf{V}} & \tilde{\mathbf{V}}\\ \tilde{\mathbf{U}} & -\tilde{\mathbf{U}}\end{bmatrix}^T\begin{bmatrix}\mathbf{V}\sqrt{\mathbf{S}}\mathbf{R}^T\\ \mathbf{U}\sqrt{\mathbf{S}}\mathbf{R}^T\end{bmatrix}\\
&= \frac{1}{\sqrt{2}}\begin{bmatrix}\tilde{\mathbf{V}}^T\mathbf{V}\sqrt{\mathbf{S}}\mathbf{R}^T + \tilde{\mathbf{U}}^T\mathbf{U}\sqrt{\mathbf{S}}\mathbf{R}^T\\ \tilde{\mathbf{V}}^T\mathbf{V}\sqrt{\mathbf{S}}\mathbf{R}^T - \tilde{\mathbf{U}}^T\mathbf{U}\sqrt{\mathbf{S}}\mathbf{R}^T\end{bmatrix}\\
&= \frac{1}{\sqrt{2}}\begin{bmatrix}\left(\tilde{\mathbf{V}}^T\mathbf{V}+\tilde{\mathbf{U}}^T\mathbf{U}\right)\sqrt{\mathbf{S}}\mathbf{R}^T\\ \left(\tilde{\mathbf{V}}^T\mathbf{V}-\tilde{\mathbf{U}}^T\mathbf{U}\right)\sqrt{\mathbf{S}}\mathbf{R}^T\end{bmatrix},
\end{aligned} \tag{A.94}$$

such that

$$\begin{aligned}
\mathbf{O}e^{\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{O}^T\mathbf{Q}(0) &= \frac{1}{2}\begin{bmatrix}\tilde{\mathbf{V}}e^{\tilde{\mathbf{S}}\frac{t}{\tau}} & \tilde{\mathbf{V}}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\\ \tilde{\mathbf{U}}e^{\tilde{\mathbf{S}}\frac{t}{\tau}} & -\tilde{\mathbf{U}}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\end{bmatrix}\begin{bmatrix}\tilde{\mathbf{V}}^T\mathbf{V}\sqrt{\mathbf{S}}\mathbf{R}^T + \tilde{\mathbf{U}}^T\mathbf{U}\sqrt{\mathbf{S}}\mathbf{R}^T\\ \tilde{\mathbf{V}}^T\mathbf{V}\sqrt{\mathbf{S}}\mathbf{R}^T - \tilde{\mathbf{U}}^T\mathbf{U}\sqrt{\mathbf{S}}\mathbf{R}^T\end{bmatrix}\\
&= \frac{1}{2}\begin{bmatrix}\tilde{\mathbf{V}}\left(e^{\tilde{\mathbf{S}}\frac{t}{\tau}}\left(\tilde{\mathbf{V}}^T\mathbf{V}+\tilde{\mathbf{U}}^T\mathbf{U}\right)+e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\left(\tilde{\mathbf{V}}^T\mathbf{V}-\tilde{\mathbf{U}}^T\mathbf{U}\right)\right)\sqrt{\mathbf{S}}\mathbf{R}^T\\ \tilde{\mathbf{U}}\left(e^{\tilde{\mathbf{S}}\frac{t}{\tau}}\left(\tilde{\mathbf{V}}^T\mathbf{V}+\tilde{\mathbf{U}}^T\mathbf{U}\right)-e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\left(\tilde{\mathbf{V}}^T\mathbf{V}-\tilde{\mathbf{U}}^T\mathbf{U}\right)\right)\sqrt{\mathbf{S}}\mathbf{R}^T\end{bmatrix}.
\end{aligned} \tag{A.95}$$

We continue by calculating

$$\begin{aligned}
4\mathbf{M}\mathbf{M}^T\mathbf{Q}(0) &= 4\frac{1}{\sqrt{2}}\begin{bmatrix}\tilde{\mathbf{V}}_\perp\\ \tilde{\mathbf{U}}_\perp\end{bmatrix}\frac{1}{\sqrt{2}}\begin{bmatrix}\tilde{\mathbf{V}}_\perp\\ \tilde{\mathbf{U}}_\perp\end{bmatrix}^T\begin{bmatrix}\mathbf{V}\sqrt{\mathbf{S}}\mathbf{R}^T\\ \mathbf{U}\sqrt{\mathbf{S}}\mathbf{R}^T\end{bmatrix}\\
&= 2\begin{bmatrix}\tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T & \tilde{\mathbf{V}}_\perp\tilde{\mathbf{U}}_\perp^T\\ \tilde{\mathbf{U}}_\perp\tilde{\mathbf{V}}_\perp^T & \tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\end{bmatrix}\begin{bmatrix}\mathbf{V}\sqrt{\mathbf{S}}\mathbf{R}^T\\ \mathbf{U}\sqrt{\mathbf{S}}\mathbf{R}^T\end{bmatrix}\\
&= 2\begin{bmatrix}\tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T & 0\\ 0 & \tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\end{bmatrix}\begin{bmatrix}\mathbf{V}\sqrt{\mathbf{S}}\mathbf{R}^T\\ \mathbf{U}\sqrt{\mathbf{S}}\mathbf{R}^T\end{bmatrix}\\
&= 2\begin{bmatrix}\tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T\mathbf{V}\sqrt{\mathbf{S}}\mathbf{R}^T\\ \tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\mathbf{U}\sqrt{\mathbf{S}}\mathbf{R}^T\end{bmatrix}
\end{aligned} \tag{A.96}$$

and

$$\begin{aligned}\frac{1}{2}\mathbf{Q}(0)^T 4\frac{t}{\tau}\mathbf{M}\mathbf{M}^T\mathbf{Q}(0) &= \frac{t}{\tau}\begin{bmatrix}\mathbf{R}\sqrt{\mathbf{S}}\mathbf{V}^T & \mathbf{R}\sqrt{\mathbf{S}}\mathbf{U}^T\end{bmatrix}\begin{bmatrix}\tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T\mathbf{V}\sqrt{\mathbf{S}}\mathbf{R}^T \\ \tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\mathbf{U}\sqrt{\mathbf{S}}\mathbf{R}^T\end{bmatrix} \\ &= \frac{t}{\tau}\left[\mathbf{R}\sqrt{\mathbf{S}}\left(\mathbf{V}^T\tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T\mathbf{V} + \mathbf{U}^T\tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\mathbf{U}\right)\sqrt{\mathbf{S}}\mathbf{R}^T\right]. \qquad (A.97)\end{aligned}$$

Next, we define $\mathbf{B}_0 = \mathbf{U}^T\tilde{\mathbf{U}} + \mathbf{V}^T\tilde{\mathbf{V}}$ and $\mathbf{C}_0 = \mathbf{U}^T\tilde{\mathbf{U}} - \mathbf{V}^T\tilde{\mathbf{V}}$ and rewrite the inverse as

$$\begin{aligned}&\left[\mathbf{I} + \frac{1}{2}\mathbf{Q}(0)^T\mathbf{O}\left(e^{2\mathbf{\Lambda}\frac{t}{\tau}} - \mathbf{I}\right)\mathbf{\Lambda}^{-1}\mathbf{O}^T\mathbf{Q}(0) + 2\frac{t}{\tau}\mathbf{Q}(0)^T\mathbf{M}\mathbf{M}^T\mathbf{Q}(0)\right]^{-1} \\ &= \left[\mathbf{I} + \frac{1}{4}\mathbf{R}\sqrt{\mathbf{S}}\left(\begin{bmatrix}\mathbf{B}_0 & -\mathbf{C}_0\end{bmatrix}\left(e^{2\mathbf{\Lambda}\frac{t}{\tau}} - \mathbf{I}\right)\mathbf{\Lambda}^{-1}\begin{bmatrix}\mathbf{B}_0^T \\ -\mathbf{C}_0^T\end{bmatrix}\right.\right. \\ &\quad \left.\left. + 4\frac{t}{\tau}\left(\mathbf{V}^T\tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T\mathbf{V} + \mathbf{U}^T\tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\mathbf{U}\right)\right)\sqrt{\mathbf{S}}\mathbf{R}^T\right]^{-1}. \qquad (A.98)\end{aligned}$$

Working from the centre out, we have

$$\begin{aligned}\begin{bmatrix}\mathbf{B}_0 & -\mathbf{C}_0\end{bmatrix}\mathbf{\Lambda}^{-1}\begin{bmatrix}\mathbf{B}_0^T \\ -\mathbf{C}_0^T\end{bmatrix} &= \begin{bmatrix}\mathbf{B}_0 & -\mathbf{C}_0\end{bmatrix}\begin{bmatrix}\tilde{\mathbf{S}}^{-1} & 0 \\ 0 & -\tilde{\mathbf{S}}^{-1}\end{bmatrix}\begin{bmatrix}\mathbf{B}_0^T \\ -\mathbf{C}_0^T\end{bmatrix} \\ &= \begin{bmatrix}\mathbf{B}_0 & -\mathbf{C}_0\end{bmatrix}\begin{bmatrix}\tilde{\mathbf{S}}^{-1}\mathbf{B}_0^T \\ \tilde{\mathbf{S}}^{-1}\mathbf{C}_0^T\end{bmatrix} \\ &= \mathbf{B}_0\tilde{\mathbf{S}}^{-1}\mathbf{B}_0^T - \mathbf{C}_0\tilde{\mathbf{S}}^{-1}\mathbf{C}_0^T \qquad (A.99)\end{aligned}$$

and

$$\begin{aligned}\begin{bmatrix}\mathbf{B}_0 & -\mathbf{C}_0\end{bmatrix}e^{2\mathbf{\Lambda}\frac{t}{\tau}}\mathbf{\Lambda}^{-1}\begin{bmatrix}\mathbf{B}_0^T \\ -\mathbf{C}_0^T\end{bmatrix} &= \begin{bmatrix}\mathbf{B}_0 & -\mathbf{C}_0\end{bmatrix}\begin{bmatrix}e^{2\tilde{\mathbf{S}}\frac{t}{\tau}}\tilde{\mathbf{S}}^{-1} & 0 \\ 0 & -e^{-2\tilde{\mathbf{S}}\frac{t}{\tau}}\tilde{\mathbf{S}}^{-1}\end{bmatrix}\begin{bmatrix}\mathbf{B}_0^T \\ -\mathbf{C}_0^T\end{bmatrix} \\ &= \begin{bmatrix}\mathbf{B}_0 & -\mathbf{C}_0\end{bmatrix}\begin{bmatrix}e^{2\tilde{\mathbf{S}}\frac{t}{\tau}}\tilde{\mathbf{S}}^{-1}\mathbf{B}_0^T \\ e^{-2\tilde{\mathbf{S}}\frac{t}{\tau}}\tilde{\mathbf{S}}^{-1}\mathbf{C}_0^T\end{bmatrix} \\ &= \mathbf{B}_0 e^{2\tilde{\mathbf{S}}\frac{t}{\tau}}\tilde{\mathbf{S}}^{-1}\mathbf{B}_0^T - \mathbf{C}_0 e^{-2\tilde{\mathbf{S}}\frac{t}{\tau}}\tilde{\mathbf{S}}^{-1}\mathbf{C}_0^T. \qquad (A.100)\end{aligned}$$

In this setting, we require that $\mathbf{B_0}$ is invertible. Finally, using $\mathbf{AB}^{-1} = (\mathbf{BA}^{-1})^{-1}$ (and $\mathbf{A}^{-1}\mathbf{B} = (\mathbf{B}^{-1}\mathbf{A})^{-1}$) to move terms into the inverse, we rewrite

$$
\begin{aligned}
\mathbf{QQ}^T(t) =& \frac{1}{2}\begin{bmatrix} \left(\tilde{\mathbf{V}}\left(e^{\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{B}_0^T - e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{C}_0^T\right) + 2\tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T\mathbf{V}\right)\sqrt{\mathbf{S}}\mathbf{R}^T \\ \left(\tilde{\mathbf{U}}\left(e^{\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{B}_0^T + e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{C}_0^T\right) + 2\tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\mathbf{U}\right)\sqrt{\mathbf{S}}\mathbf{R}^T \end{bmatrix} \\
& \Bigg[\mathbf{I} + \mathbf{R}\sqrt{\mathbf{S}}\Bigg(\frac{1}{4}\mathbf{B}_0\left(e^{2\tilde{\mathbf{S}}\frac{t}{\tau}} - \mathbf{I}\right)\tilde{\mathbf{S}}^{-1}\mathbf{B}_0^T - \frac{1}{4}\mathbf{C}_0\left(e^{-2\tilde{\mathbf{S}}\frac{t}{\tau}} - \mathbf{I}\right)\tilde{\mathbf{S}}^{-1}\mathbf{C}_0^T \\
& + \frac{t}{\tau}\left(\mathbf{V}^T\tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T\mathbf{V} + \mathbf{U}^T\tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\mathbf{U}\right)\Bigg)\sqrt{\mathbf{S}}\mathbf{R}^T\Bigg]^{-1} \\
& \frac{1}{2}\begin{bmatrix} \left(\tilde{\mathbf{V}}\left(e^{\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{B}_0^T - e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{C}_0^T\right) + 2\tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T\mathbf{V}\right)\sqrt{\mathbf{S}}\mathbf{R}^T \\ \left(\tilde{\mathbf{U}}\left(e^{\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{B}_0^T + e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{C}_0^T\right) + 2\tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\mathbf{U}\right)\sqrt{\mathbf{S}}\mathbf{R}^T \end{bmatrix}^T \\
=& \frac{1}{2}\begin{bmatrix} \tilde{\mathbf{V}}\left(e^{\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{B}_0^T - e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{C}_0^T\right) + 2\tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T\mathbf{V} \\ \tilde{\mathbf{U}}\left(e^{\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{B}_0^T + e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{C}_0^T\right) + 2\tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\mathbf{U} \end{bmatrix} \\
& \Bigg[\mathbf{S}^{-1} + \frac{1}{4}\mathbf{B}_0\left(e^{2\tilde{\mathbf{S}}\frac{t}{\tau}} - \mathbf{I}\right)\tilde{\mathbf{S}}^{-1}\mathbf{B}_0^T - \frac{1}{4}\mathbf{C}_0\left(e^{-2\tilde{\mathbf{S}}\frac{t}{\tau}} - \mathbf{I}\right)\tilde{\mathbf{S}}^{-1}\mathbf{C}_0^T \\
& + \frac{t}{\tau}\left(\mathbf{V}^T\tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T\mathbf{V} + \mathbf{U}^T\tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\mathbf{U}\right)\Bigg]^{-1} \\
& \frac{1}{2}\begin{bmatrix} \tilde{\mathbf{V}}\left(e^{\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{B}_0^T - e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{C}_0^T\right) + 2\tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T\mathbf{V} \\ \tilde{\mathbf{U}}\left(e^{\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{B}_0^T + e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{C}_0^T\right) + 2\tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\mathbf{U} \end{bmatrix}^T \\
=& \begin{bmatrix} \tilde{\mathbf{V}}\left(\mathbf{I} - e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{C}_0^T(\mathbf{B}_0^T)^{-1}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\right) + 2\tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T\mathbf{V}(\mathbf{B}_0^T)^{-1}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}} \\ \tilde{\mathbf{U}}\left(\mathbf{I} + e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{C}_0^T(\mathbf{B}_0^T)^{-1}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\right) + 2\tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\mathbf{U}(\mathbf{B}_0^T)^{-1}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}} \end{bmatrix} \\
& \Bigg[4e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{B}_0^{-1}\mathbf{S}^{-1}(\mathbf{B}_0^T)^{-1}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}} + \left(\mathbf{I} - e^{-2\tilde{\mathbf{S}}\frac{t}{\tau}}\right)\tilde{\mathbf{S}}^{-1} \\
& - e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{B}_0^{-1}\mathbf{C}_0\left(e^{-2\tilde{\mathbf{S}}\frac{t}{\tau}} - \mathbf{I}\right)\tilde{\mathbf{S}}^{-1}\mathbf{C}_0^T(\mathbf{B}_0^T)^{-1}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}} \qquad \text{(A.101)} \\
& + 4\frac{t}{\tau}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{B}_0^{-1}\left(\mathbf{V}^T\tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T\mathbf{V} + \mathbf{U}^T\tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\mathbf{U}\right)(\mathbf{B}_0^T)^{-1}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\Bigg]^{-1} \\
& \begin{bmatrix} \tilde{\mathbf{V}}\left(\mathbf{I} - e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{C}_0^T(\mathbf{B}_0^T)^{-1}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\right) + 2\tilde{\mathbf{V}}_\perp\tilde{\mathbf{V}}_\perp^T\mathbf{V}B^{-T}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}} \\ \tilde{\mathbf{U}}\left(\mathbf{I} + e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{C}_0^T(\mathbf{B}_0^T)^{-1}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\right) + 2\tilde{\mathbf{U}}_\perp\tilde{\mathbf{U}}_\perp^T\mathbf{U}B^{-T}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}} \end{bmatrix}^T .
\end{aligned}
$$

A.3 Rich-Lazy learning

A.3.1 Dynamics of the singular values

Theorem A.3.1. *Under the assumptions of Theorem 3.3.3 and with a task-aligned initialisation given by* $\mathbf{W}_1(0) = \mathbf{R}\mathbf{S}_1\tilde{\mathbf{V}}^T$ *and* $\mathbf{W}_2(0) = \tilde{\mathbf{U}}\mathbf{S}_2\mathbf{R}^T$*, where* $\mathbf{R} \in \mathbb{R}^{N_h \times N_h}$ *is an orthonormal matrix, then the network function is given by the expression* $\mathbf{W}_2\mathbf{W}_1(t) = \tilde{\mathbf{U}}\mathbf{S}(t)\tilde{\mathbf{V}}^T$ *where* $\mathbf{S}(t) \in \mathbb{R}^{N_h \times N_h}$ *is a diagonal matrix of singular values with elements* $s_\alpha(t)$ *that evolve according to the equation,*

$$s_\alpha(t) = s_\alpha(0) + \gamma_\alpha(t;\lambda)\left(\tilde{s}_\alpha - s_\alpha(0)\right), \tag{A.102}$$

where $\tilde{s}_\alpha$ *is the* α *singular value of* $\tilde{\mathbf{S}}$ *and* $\gamma_\alpha(t;\lambda)$ *is a* λ*-dependent monotonic transition function for each singular value that increases from* $\gamma_\alpha(0;\lambda) = 0$ *to* $\lim_{t\to\infty}\gamma_\alpha(t;\lambda) = 1$ *defined as*

$$\gamma_\alpha(t;\lambda) = \frac{\tilde{s}_{\lambda,\alpha}s_{\lambda,\alpha}\sinh\left(2\tilde{s}_{\lambda,\alpha}\frac{t}{\tau}\right) + \left(\tilde{s}_\alpha s_\alpha + \frac{\lambda^2}{4}\right)\cosh\left(2\tilde{s}_{\lambda,\alpha}\frac{t}{\tau}\right) - \left(\tilde{s}_\alpha s_\alpha + \frac{\lambda^2}{4}\right)}{\tilde{s}_{\lambda,\alpha}s_{\lambda,\alpha}\sinh\left(2\tilde{s}_{\lambda,\alpha}\frac{t}{\tau}\right) + \left(\tilde{s}_\alpha s_\alpha + \frac{\lambda^2}{4}\right)\cosh\left(2\tilde{s}_{\lambda,\alpha}\frac{t}{\tau}\right) + \tilde{s}_\alpha\left(\tilde{s}_\alpha - s_\alpha\right)}, \tag{A.103}$$

where $\tilde{s}_{\lambda,\alpha} = \sqrt{\tilde{s}_\alpha^2 + \frac{\lambda^2}{4}}$, $s_{\lambda,\alpha} = \sqrt{s_\alpha(0)^2 + \frac{\lambda^2}{4}}$, *and* $s_\alpha = s_\alpha(0)$. *We find that under different limits of* λ*, the transition function converges pointwise to the sigmoidal (*$\lambda \to 0$*) and exponential (*$\lambda \to \pm\infty$*) transition functions,*

$$\gamma_\alpha(t;\lambda) \to \begin{cases} \frac{e^{2\tilde{s}_\alpha\frac{t}{\tau}}-1}{e^{2\tilde{s}_\alpha\frac{t}{\tau}}-1+\frac{\tilde{s}_\alpha}{s_\alpha(0)}} & \text{as } \lambda \to 0, \\ 1 - e^{-|\lambda|\frac{t}{\tau}} & \text{as } \lambda \to \pm\infty \end{cases}. \tag{A.104}$$

Proof. According to Theorem 3.3.3, the network function is given by the equation

$$\mathbf{W}_2\mathbf{W}_1(t) = \mathbf{Z}_2(t)\mathbf{A}^{-1}(t)\mathbf{Z}_1^T(t), \tag{A.105}$$

which depends on the variables of the initialisation $\mathbf{B}$ and $\mathbf{C}$. Plugging the expressions for a task-aligned initialisation $\mathbf{W}_1(0)$ and $\mathbf{W}_2(0)$ into these variables we get the following simplified expressions,

$$\mathbf{B} = \mathbf{R}\underbrace{\left(\mathbf{S}_2(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}}) + \mathbf{S}_1(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}})\right)}_{\mathbf{D}_B}, \tag{A.106}$$

$$\mathbf{C} = \mathbf{R}\underbrace{\left(\mathbf{S}_2(\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}}) - \mathbf{S}_1(\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}})\right)}_{\mathbf{D}_C}, \tag{A.107}$$

where we define the diagonal matrices $\mathbf{D}_B$ and $\mathbf{D}_C$ for ease of notation. Using these expressions, we now get the following time-dependent expressions for $\mathbf{Z}_2(t)$, $\mathbf{A}^{-1}(t)$, and $\mathbf{Z}_1(t)$,

$$\mathbf{Z}_1(t) = \frac{1}{2}\tilde{\mathbf{V}}\left((\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{D}_B - (\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{D}_C\right)\mathbf{R}^T. \tag{A.108}$$

$$\mathbf{Z}_2(t) = \frac{1}{2}\tilde{\mathbf{U}}\left((\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{D}_B + (\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{D}_C\right)\mathbf{R}^T. \tag{A.109}$$

$$\mathbf{A}(t) = \mathbf{R}\left(\mathbf{I} + \left(\frac{e^{2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} - \mathbf{I}}{4\tilde{\mathbf{S}}_\lambda}\right)\mathbf{D}_B^2 - \left(\frac{e^{-2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} - \mathbf{I}}{4\tilde{\mathbf{S}}_\lambda}\right)\mathbf{D}_C^2\right)\mathbf{R}^T. \tag{A.110}$$

Plugging these expressions into the expression for the network function, notice that the $\mathbf{R}$ terms cancel each other resulting in following equation

$$\mathbf{W}_2\mathbf{W}_1(t) = \tilde{\mathbf{U}}\underbrace{\left(\frac{\mathbf{N_{um}}}{4\mathbf{I} + \left(\frac{e^{2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} - \mathbf{I}}{\tilde{\mathbf{S}}_\lambda}\right)\mathbf{D}_B^2 - \left(\frac{e^{-2\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}} - \mathbf{I}}{\tilde{\mathbf{S}}_\lambda}\right)\mathbf{D}_C^2}\right)}_{\mathbf{S}(t)}\tilde{\mathbf{V}}^T, \tag{A.111}$$

where

$$\mathbf{N_{um}} = \left((\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{D}_B - (\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{D}_C\right) \times \left((\tilde{\mathbf{G}} + \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{D}_B + (\tilde{\mathbf{G}} - \tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda \frac{t}{\tau}}\mathbf{D}_C\right) \tag{A.112}$$

Notice that the middle term is simply a product of diagonal matrices. We can factor the numerator of this expressions as,

$$(\tilde{\mathbf{G}}^2-\tilde{\mathbf{H}}^2\tilde{\mathbf{G}}^2)e^{2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_B^2+\left((\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})^2-(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})^2\right)\mathbf{D}_B\mathbf{D}_C-(\tilde{\mathbf{G}}^2-\tilde{\mathbf{H}}^2\tilde{\mathbf{G}}^2)e^{-2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_C^2 \tag{A.113}$$

We can further factor this expression as,

$$\tilde{\mathbf{G}}^2(\mathbf{I}-\tilde{\mathbf{H}}^2)\left(e^{2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_B^2-e^{-2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_C^2\right)-4\tilde{\mathbf{G}}^2\tilde{\mathbf{H}}\mathbf{D}_B\mathbf{D}_C. \tag{A.114}$$

Putting it all together we find that $\mathbf{S}(t)$ can be expressed as,

$$\mathbf{S}(t)=\frac{\tilde{\mathbf{G}}^2(\mathbf{I}-\tilde{\mathbf{H}}^2)\left(e^{2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_B^2-e^{-2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_C^2\right)-4\tilde{\mathbf{G}}^2\tilde{\mathbf{H}}\mathbf{D}_B\mathbf{D}_C}{4\mathbf{I}+\left(\frac{e^{2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}-\mathbf{I}}{\tilde{\mathbf{S}}_\lambda}\right)\mathbf{D}_B^2-\left(\frac{e^{-2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}-\mathbf{I}}{\tilde{\mathbf{S}}_\lambda}\right)\mathbf{D}_C^2}. \tag{A.115}$$

Now using the relationship between $\tilde{\mathbf{H}}$ and $\tilde{\mathbf{G}}$ we use the following two identities:

$$\tilde{\mathbf{G}}^2(\mathbf{I}-\tilde{\mathbf{H}}^2)=\frac{\tilde{\mathbf{S}}}{\tilde{\mathbf{S}}_\lambda},\qquad 4\tilde{\mathbf{G}}^2\tilde{\mathbf{H}}=\frac{\lambda}{\tilde{\mathbf{S}}_\lambda} \tag{A.116}$$

Plugging these identities into the previous expression and multiplying the numerator and denominator by $\tilde{\mathbf{S}}_\lambda$ gives,

$$\mathbf{S}(t)=\frac{\tilde{\mathbf{S}}\left(e^{2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_B^2-e^{-2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_C^2\right)-\lambda\mathbf{D}_B\mathbf{D}_C}{4\tilde{\mathbf{S}}_\lambda+e^{2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_B^2-e^{-2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_C^2+\mathbf{D}_C^2-\mathbf{D}_B^2}. \tag{A.117}$$

Add and subtract $\tilde{\mathbf{S}}\left(4\tilde{\mathbf{S}}_\lambda+\mathbf{D}_C^2-\mathbf{D}_B^2\right)$ from the numerator such that

$$\mathbf{S}(t)=\tilde{\mathbf{S}}-\frac{\tilde{\mathbf{S}}\left(4\tilde{\mathbf{S}}_\lambda+\mathbf{D}_C^2-\mathbf{D}_B^2\right)+\lambda\mathbf{D}_B\mathbf{D}_C}{4\tilde{\mathbf{S}}_\lambda+e^{2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_B^2-e^{-2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_C^2+\mathbf{D}_C^2-\mathbf{D}_B^2}. \tag{A.118}$$

Using the form of $\mathbf{D}_B$ and $\mathbf{D}_C$ notice the following two identities:

$$\mathbf{D}_B\mathbf{D}_C=\frac{\lambda}{\tilde{\mathbf{S}}_\lambda}\left(\tilde{\mathbf{S}}-\mathbf{S}_2\mathbf{S}_1\right),\qquad \mathbf{D}_C^2-\mathbf{D}_B^2=-\frac{4}{\tilde{\mathbf{S}}_\lambda}\left(\tilde{\mathbf{S}}\mathbf{S}_2\mathbf{S}_1+\frac{\lambda^2}{4}\mathbf{I}\right) \tag{A.119}$$

From the second identity we can derive a third identity,

$$4\tilde{\mathbf{S}}_\lambda + \mathbf{D}_C^2 - \mathbf{D}_B^2 = 4\frac{\tilde{\mathbf{S}}}{\tilde{\mathbf{S}}_\lambda}\left(\tilde{\mathbf{S}} - \mathbf{S}_2\mathbf{S}_1\right) \tag{A.120}$$

Plugging the first and third identities into the numerator for the previous expression gives,

$$\mathbf{S}(t) = \tilde{\mathbf{S}} - \frac{\frac{\left(4\tilde{\mathbf{S}}^2 + \lambda^2\mathbf{I}\right)}{\tilde{\mathbf{S}}_\lambda}\left(\tilde{\mathbf{S}} - \mathbf{S}_2\mathbf{S}_1\right)}{4\tilde{\mathbf{S}}_\lambda + e^{2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_B^2 - e^{-2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_C^2 + \mathbf{D}_C^2 - \mathbf{D}_B^2}. \tag{A.121}$$

Multiply numerator and denominator by $\frac{\tilde{\mathbf{S}}_\lambda}{4}$ and simplify terms gives the expression,

$$\mathbf{S}(t) = \tilde{\mathbf{S}} - \frac{\tilde{\mathbf{S}}_\lambda^2}{\tilde{\mathbf{S}}_\lambda^2 + \frac{\tilde{\mathbf{S}}_\lambda}{4}\left(e^{2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_B^2 - e^{-2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_C^2\right) - \frac{\tilde{\mathbf{S}}_\lambda}{4}\left(\mathbf{D}_B^2 - \mathbf{D}_C^2\right)}\left(\tilde{\mathbf{S}} - \mathbf{S}_2\mathbf{S}_1\right). \tag{A.122}$$

Thus we have found the transition function,

$$\gamma(t;\lambda) = \frac{\frac{\tilde{\mathbf{S}}_\lambda}{4}\left(e^{2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_B^2 - e^{-2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_C^2\right) + \frac{\tilde{\mathbf{S}}_\lambda}{4}\left(\mathbf{D}_C^2 - \mathbf{D}_B^2\right)}{\frac{\tilde{\mathbf{S}}_\lambda}{4}\left(e^{2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_B^2 - e^{-2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_C^2\right) + \frac{\tilde{\mathbf{S}}_\lambda}{4}\left(4\tilde{\mathbf{S}}_\lambda + \mathbf{D}_C^2 - \mathbf{D}_B^2\right)}. \tag{A.123}$$

We will use our previous identities and the definitions of $\mathbf{D}_B^2$ and $\mathbf{D}_C^2$ to simplify this expression. Notice the following identity,

$$\frac{\tilde{\mathbf{S}}_\lambda}{4}\left(e^{2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_B^2 - e^{-2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{D}_C^2\right) = \tilde{\mathbf{S}}_\lambda\mathbf{S}_\lambda\sinh\left(2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}\right) + \left(\tilde{\mathbf{S}}\mathbf{S}(0) + \frac{\lambda^2}{4}\mathbf{I}\right)\cosh\left(2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}\right) \tag{A.124}$$

Putting it all together we get

$$\gamma(t;\lambda) = \frac{\tilde{\mathbf{S}}_\lambda\mathbf{S}_\lambda\sinh\left(2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}\right) + \left(\tilde{\mathbf{S}}\mathbf{S}(0) + \frac{\lambda^2}{4}\mathbf{I}\right)\cosh\left(2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}\right) - \left(\tilde{\mathbf{S}}\mathbf{S}(0) + \frac{\lambda^2}{4}\mathbf{I}\right)}{\tilde{\mathbf{S}}_\lambda\mathbf{S}_\lambda\sinh\left(2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}\right) + \left(\tilde{\mathbf{S}}\mathbf{S}(0) + \frac{\lambda^2}{4}\mathbf{I}\right)\cosh\left(2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}\right) + \tilde{\mathbf{S}}\left(\tilde{\mathbf{S}} - \mathbf{S}(0)\right)} \tag{A.125}$$

We will now show why under certain limits of λ this expression simplifies to the sigmoidal and exponential dynamics discussed in the previous section.

Sigmoidal dynamics. When $\lambda = 0$, then $\tilde{\mathbf{S}}_\lambda = \tilde{\mathbf{S}}$ and $\mathbf{S}_\lambda = \mathbf{S}(0)$. Notice, that the coefficients for the hyperbolic functions all simplify to $\tilde{\mathbf{S}}\mathbf{S}(0)$. Using the hyperbolic identity $\sinh(x)+\cosh(x)=e^x$, we can simplify the expression for the transition function to

$$\gamma(t;\lambda)=\frac{\tilde{\mathbf{S}}\mathbf{S}(0)e^{2\tilde{\mathbf{S}}\frac{t}{\tau}}-\tilde{\mathbf{S}}\mathbf{S}(0)}{\tilde{\mathbf{S}}\mathbf{S}(0)e^{2\tilde{\mathbf{S}}\frac{t}{\tau}}-\tilde{\mathbf{S}}\mathbf{S}(0)+\tilde{\mathbf{S}}^2}. \tag{A.126}$$

Dividing the numerator and denominator by $\tilde{\mathbf{S}}\mathbf{S}(0)$ gives the final expression.

Exponential dynamics. In the limit as $\lambda\to\pm\infty$ the expressions $\tilde{\mathbf{S}}_\lambda\to\frac{|\lambda|}{2}$ and $\mathbf{S}_\lambda\to\frac{|\lambda|}{2}$. Additionally, in these limits because $\frac{\lambda^2}{4}\mathbf{I}\gg\tilde{\mathbf{S}}\mathbf{S}(0)$ then $\left(\tilde{\mathbf{S}}\mathbf{S}(0)+\frac{\lambda^2}{4}\mathbf{I}\right)\to\frac{\lambda^2}{4}\mathbf{I}$. As a result of these simplifications the coefficients for the hyperbolic functions all simplify to $\frac{\lambda^2}{4}\mathbf{I}$. As a result we can again use the hyperbolic identity $\sinh(x)+\cosh(x)=e^x$ to simplify the expression as

$$\gamma(t;\lambda)=\frac{\frac{\lambda^2}{4}e^{|\lambda|\frac{t}{\tau}}-\frac{\lambda^2}{4}\mathbf{I}}{\frac{\lambda^2}{4}e^{|\lambda|\frac{t}{\tau}}+\tilde{\mathbf{S}}\left(\tilde{\mathbf{S}}-\mathbf{S}(0)\right)}. \tag{A.127}$$

Dividing the numerator and denominator by $\frac{\lambda^2}{4}$ results in all terms without a coefficient proportional to λ^2 vanishing, which simplifying further gives the final expression. □

A.3.2 Dynamics of the representation

The *lazy* and *rich* regimes are defined by the dynamics of the NTK of the network. *Lazy* learning occurs when the NTK is constant, *rich* learning occurs when it is not [87].

The NTK intuitively measures the movement of the network representations through training. As shown in our first work [39], in specific experimental setup, we can calculate the NTK of the network in terms of the internal representations in a straightforward way:

$$\text{NTK}=\mathbf{I}_{N_o}\otimes\mathbf{X}^T\mathbf{W}_1^T\mathbf{W}_1(t)\mathbf{X}+\mathbf{W}_2\mathbf{W}_2^T(t)\otimes\mathbf{X}^T\mathbf{X}. \tag{A.128}$$

In order to better understand the effect of λ on NTK dynamics, we first derive the singular values of the λ*-balanced* weights, and the representations of a λ*-balanced* network.

A.3.2.1 Lambda-balanced singular value

Theorem A.3.2. *Under a λ-Balanced initialisation (Assumption 2), if the network function $\mathbf{W}_2\mathbf{W}_1(t) = \mathbf{U}(t)\mathbf{S}(t)\mathbf{V}^T(t)$ is full rank and we define $\mathbf{S}_\lambda(t) = \sqrt{\mathbf{S}^2(t) + \frac{\lambda^2}{4}\mathbf{I}}.$, then we can recover the parameters $\mathbf{W}_2(t) = \mathbf{U}(t)\mathbf{S}_2(t)\mathbf{R}^T(t)$, $\mathbf{W}_1(t) = \mathbf{R}(t)\mathbf{S}_1(t)\mathbf{V}^T(t)$ up to time-dependent orthogonal transformation* $\mathbf{R}$*(t) of size $N_h \times N_h$, where*

$$\mathbf{S}_1(t) = \left(\left(\mathbf{S}_\lambda(t) - \frac{\lambda\mathbf{I}}{2}\right)^{\frac{1}{2}} \quad 0_{\max(0,N_i-N_o)}\right). \tag{A.129}$$

$$\mathbf{S}_2(t) = \left(\left(\mathbf{S}_\lambda(t) + \frac{\lambda\mathbf{I}}{2}\right)^{\frac{1}{2}} 0_{\max(0,N_o-N_i)}\right). \tag{A.130}$$

Proof. We prove the case $N_i \leq N_o$ and $N_h = min(N_i, N_o)$. The proof for $N_o \leq N_i$ follows the same structure. Let $\mathbf{USV}^T = \mathbf{W}_2(t)\mathbf{W}_1(t)$ be the singular value decomposition of the product of the weights at training step t. We will use $\mathbf{W}_2 = \mathbf{W}_2(t), \mathbf{W}_1 = \mathbf{W}_1(t)$ as a shorthand.

We write the initialisation for our setting $\mathbf{W}_2\mathbf{W}_1 = \mathbf{USV}^T$. We therefore can write without loss of generality the weight matrices $\mathbf{W}_2 = \mathbf{US}_2\mathbf{G}_2$ and $\mathbf{W}_1 = \mathbf{G}_1\mathbf{S}_1\mathbf{V}^T$. In this case we requiere that $\mathbf{G}_2 = \mathbf{G}_1^{-1}$ and $\mathbf{S}_1\mathbf{S}_2 = \mathbf{S}$.

We assume the balanced property such that $\mathbf{W}_2^T\mathbf{W}_2 - \mathbf{W}_1\mathbf{W}_1^T = \lambda\mathbf{I}$. We know this holds for any t since this is a conserved quantity in linear networks. The matrices $\mathbf{W}_1\mathbf{W}_1^T$ and $\mathbf{W}_2^T\mathbf{W}_2$ are symmetric, which consequently implies that their singular vectors are orthogonal. Consequently, in our specific scenario, we require that $\mathbf{G}_1$ and $\mathbf{G}_2$ are orthogonal matrices and it follows that

$$\mathbf{G}_2 = \mathbf{G}_1^{-1} = \mathbf{G}_1^T \tag{A.131}$$

$$\mathbf{G}_2 = \mathbf{G}_1^T = \mathbf{R}^T. \tag{A.132}$$

We can write $\mathbf{W}_2 = \mathbf{US}_2\mathbf{R}^T, \mathbf{W}_1 = \mathbf{RS}_1\mathbf{V}^T$, where $\mathbf{R}$ is an orthonormal matrix and $\mathbf{S}_2, \mathbf{S}_1$ are diagonal (possibly rectangular) matrices.

Hence,

$$\mathbf{R}\mathbf{S}_2^T\mathbf{S}_2\mathbf{R}^T - \mathbf{R}\mathbf{S}_1\mathbf{S}_1\mathbf{R}^T = \lambda\mathbf{I}, \tag{A.133}$$

$$\mathbf{S}_2^T\mathbf{S}_2 - \mathbf{S}_1\mathbf{S}_1 = \lambda\mathbf{I}. \tag{A.134}$$

The matrices $\mathbf{S}_1, \mathbf{S}_2$, have shapes (N_h, N_i), (N_o, N_h) respectively. We introduce the diagonal matrices $\hat{\mathbf{S}}_1$ of shape (N_h, N_i), $\hat{\mathbf{S}}_2$ of shape (N_i, N_h) such that the zero matrix has size $(N_o - N_i, N_h)$:

$$\mathbf{S}_1 = \begin{pmatrix} \hat{\mathbf{S}}_1 \end{pmatrix}, \quad \mathbf{S}_2 = \begin{pmatrix} \hat{\mathbf{S}}_2 \\ 0 \end{pmatrix}. \tag{A.135}$$

Hence,

$$\mathbf{S}_2^T\mathbf{S}_2 - \mathbf{S}_1\mathbf{S}_1 = \lambda\mathbf{I}. \tag{A.136}$$

From the above equation and the fact that $\hat{\mathbf{S}}_1\hat{\mathbf{S}}_2 = \mathbf{S}$ we derive that:

$$\hat{\mathbf{S}}_2 = \left(\frac{\sqrt{\lambda^2\mathbf{I} + 4\mathbf{S}^2} + \lambda\mathbf{I}}{2}\right)^{\frac{1}{2}}, \quad \hat{\mathbf{S}}_1 = \left(\frac{\sqrt{\lambda^2\mathbf{I} + 4\mathbf{S}^2} - \lambda\mathbf{I}}{2}\right)^{\frac{1}{2}}, \tag{A.137}$$

Hence,

$$\mathbf{W}_2 = \mathbf{U}\begin{pmatrix} \left(\frac{\sqrt{\lambda^2\mathbf{I}+4\mathbf{S}^2}+\lambda\mathbf{I}}{2}\right)^{\frac{1}{2}} \\ 0_{\max(0, N_o - N_i)} \end{pmatrix}\mathbf{R}^T, \tag{A.138}$$

$$\mathbf{W}_1 = \mathbf{R}\begin{pmatrix} \left(\frac{\sqrt{\lambda^2\mathbf{I}+4\mathbf{S}^2}-\lambda\mathbf{I}}{2}\right)^{\frac{1}{2}} & 0_{\max(0, N_i - N_o)} \end{pmatrix}\mathbf{V}^T. \tag{A.139}$$

□

A.3.2.2 Convergence proof

With our solution, $\mathbf{Q}\mathbf{Q}^T(t)$, which captures the temporal dynamics of the similarity between hidden layer activations, we can analyse the network's internal representations in relation to the task. This allows us to determine whether the network adopts a *rich* or *lazy* representation, depending on the value of λ. Consider a λ-Balanced network training on data $\mathbf{\Sigma}^{yx} = \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T$. We assume that the convergence is toward global minima and $\mathbf{B}$ is invertible

Theorem A.3.3. *Under the assumptions of Theorem A.2.6, the network function converges to* $\tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T$ *and acquires the internal representation, that is* $\mathbf{W}_1^T\mathbf{W}_1 = \tilde{\mathbf{V}}\tilde{\mathbf{S}}_1^2\tilde{\mathbf{V}}^T$ *and* $\mathbf{W}_2\mathbf{W}_2^T = \tilde{\mathbf{U}}\tilde{\mathbf{S}}_2^2\tilde{\mathbf{U}}^T$

Proof. As training time increases, all terms in Eq. A.72 vanish to zero. Terms in Eq. A.72 decay as

$$\lim_{t\to\infty} e^{-\sqrt{\tilde{\mathbf{S}}^2+\frac{\lambda^2\mathbf{I}}{4}}\frac{t}{\tau}} = \mathbf{0}. \tag{A.140}$$

and

$$\lim_{t\to\infty} e^{\lambda_\perp\frac{t}{\tau}} e^{-\sqrt{\tilde{\mathbf{S}}^2+\frac{\lambda^2}{4}\mathbf{I}}\frac{t}{\tau}} = \mathbf{0}. \tag{A.141}$$

Therefore, in the temporal limit, Eq. A.72 reduces to

$$\begin{aligned}\lim_{t\to\infty}\mathbf{Q}\mathbf{Q}^T(t) &= \lim_{t\to\infty}\begin{bmatrix}\mathbf{W}_1^T\mathbf{W}_1(t) & \mathbf{W}_1^T\mathbf{W}_2^T(t)\\ \mathbf{W}_2\mathbf{W}_1(t) & \mathbf{W}_2\mathbf{W}_2(t)^T\end{bmatrix}\\
&= \begin{bmatrix}\tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})\\ \tilde{\mathbf{U}}(\tilde{\mathbf{H}}\tilde{\mathbf{G}}+\tilde{\mathbf{G}})\end{bmatrix}\left[\tilde{\mathbf{S}}_\lambda^{-1}\right]^{-1}\left[(\tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}}))^T \quad (\tilde{\mathbf{U}}(\tilde{\mathbf{H}}\tilde{\mathbf{G}}+\tilde{\mathbf{G}}))^T\right]\\
&= \begin{bmatrix}\tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})\tilde{\mathbf{S}}_\lambda(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})^T\tilde{\mathbf{V}}^T & \tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})\tilde{\mathbf{S}}_\lambda(\tilde{\mathbf{H}}\tilde{\mathbf{G}}+\tilde{\mathbf{G}})^T\tilde{\mathbf{U}}^T\\ \tilde{\mathbf{U}}(\tilde{\mathbf{H}}\tilde{\mathbf{G}}+\tilde{\mathbf{G}})\tilde{\mathbf{S}}_\lambda(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})^T\tilde{\mathbf{V}}^T & \tilde{\mathbf{U}}(\tilde{\mathbf{H}}\tilde{\mathbf{G}}+\tilde{\mathbf{G}})\tilde{\mathbf{S}}_\lambda(\tilde{\mathbf{H}}\tilde{\mathbf{G}}+\tilde{\mathbf{G}})^T\tilde{\mathbf{U}}^T\end{bmatrix}.\end{aligned} \tag{A.142}$$

We note that

$$(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})\tilde{\mathbf{S}}_\lambda(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}}) = \frac{\mathbf{S}_\lambda(1-\tilde{\mathbf{H}}^2)}{1+\tilde{\mathbf{H}}^2} = \tilde{\mathbf{S}}, \tag{A.143}$$

and

$$\tilde{\mathbf{S}}_1^2 = \tilde{\mathbf{S}}_\lambda(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})^2 = \frac{\tilde{\mathbf{S}}_\lambda(1+\tilde{\mathbf{H}}^2)}{1+\tilde{\mathbf{H}}^2} - \frac{\tilde{\mathbf{S}}_\lambda(2\tilde{\mathbf{H}})}{1+\tilde{\mathbf{H}}^2} = \frac{\sqrt{4\tilde{\mathbf{S}}^2+\lambda^2\mathbf{I}}-\lambda\mathbf{I}}{2}, \tag{A.144}$$

$$\tilde{\mathbf{S}}_2^2 = \tilde{\mathbf{S}}_\lambda(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})^2 = \frac{\tilde{\mathbf{S}}_\lambda(1+\tilde{\mathbf{H}}^2)}{1+\tilde{\mathbf{H}}^2} + \frac{\tilde{\mathbf{S}}_\lambda(2\tilde{\mathbf{H}})}{1+\tilde{\mathbf{H}}^2} = \frac{\sqrt{4\tilde{\mathbf{S}}^2+\lambda^2\mathbf{I}}+\lambda\mathbf{I}}{2}. \tag{A.145}$$

Therefore, we can rewrite equation Eq. A.142 as

$$\begin{aligned}\lim_{t\to\infty}\mathbf{Q}\mathbf{Q}^T(t) &= \lim_{t\to\infty}\begin{bmatrix}\mathbf{W}_1^T\mathbf{W}_1(t) & \mathbf{W}_1^T\mathbf{W}_2^T(t)\\ \mathbf{W}_2\mathbf{W}_1(t) & \mathbf{W}_2\mathbf{W}_2(t)^T\end{bmatrix}\\ &= \begin{bmatrix}\tilde{\mathbf{V}}\tilde{\mathbf{S}}_1^2\tilde{\mathbf{V}}^T & \tilde{\mathbf{V}}\tilde{\mathbf{S}}\tilde{\mathbf{U}}^T\\ \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T & \tilde{\mathbf{U}}\tilde{\mathbf{S}}_2^2\tilde{\mathbf{U}}^T\end{bmatrix}.\end{aligned} \tag{A.146}$$

□

A.3.2.3 Representation in the limit

Theorem A.3.4. *Under the assumptions of Theorem A.2.6, training on data $\mathbf{\Sigma}^{yx} = \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T$, as $\lambda \to 0$, the representation tends to*

$$\lim_{t\to\infty}\mathbf{Q}\mathbf{Q}^T(t) = \begin{bmatrix}\tilde{\mathbf{V}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T & \tilde{\mathbf{V}}\tilde{\mathbf{S}}\tilde{\mathbf{U}}^T\\ \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T & \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{U}}^T\end{bmatrix}. \tag{A.147}$$

Proof. When $\lambda = 0$, both $\tilde{\mathbf{S}}_1$ and $\tilde{\mathbf{S}}_2$ reduce to $\tilde{\mathbf{S}}$. Substituting these expressions back into Eq. A.146 gives the expression in Eq. A.147. □

Theorem A.3.5. *Under the assumptions of Theorem A.2.6, training on data $\mathbf{\Sigma}^{yx} = \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T$, as $\lambda \to \infty$, the representation tends to*

$$\mathbf{W}_2\mathbf{W}_2^T = \tilde{\mathbf{U}}\begin{bmatrix}\lambda\mathbf{I} & 0_{\max(0,N_o-N_i)}\\ 0_{\max(0,N_o-N_i)} & 0\end{bmatrix}\tilde{\mathbf{U}}^T, \tag{A.148}$$

$$\mathbf{W}_1^T\mathbf{W}_1 = \frac{1}{\lambda}\tilde{\mathbf{V}}\begin{bmatrix}\tilde{\mathbf{S}}^2 & 0_{\max(0,N_i-N_o)}\\ 0_{\max(0,N_i-N_o)} & 0\end{bmatrix}\tilde{\mathbf{V}}^T. \tag{A.149}$$

As $\lambda \to -\infty$,

$$\mathbf{W}_2\mathbf{W}_2^T = -\frac{1}{\lambda}\tilde{\mathbf{U}}\begin{bmatrix}\tilde{\mathbf{S}}^2 & 0_{\max(0,N_o-N_i)}\\ 0_{\max(0,N_o-N_i)} & 0\end{bmatrix}\tilde{\mathbf{U}}^T, \tag{A.150}$$

$$\mathbf{W}_1^T \mathbf{W}_1 = \tilde{\mathbf{V}} \begin{bmatrix} -\lambda \mathbf{I} & 0_{\max(0,N_i-N_o)} \\ 0_{\max(0,N_i-N_o)} & 0 \end{bmatrix} \tilde{\mathbf{V}}^T. \tag{A.151}$$

Proof. We start from the representation derived in Theorem A.3.3 and using the Taylor expansion of $f(x) = \sqrt{1+x^2}$, we compute

$$\frac{\sqrt{\lambda^2 \mathbf{I} + 4\tilde{\mathbf{S}}^2} + \lambda \mathbf{I}}{2} = \frac{|\lambda| \sqrt{1 + \left(\frac{2\tilde{\mathbf{S}}}{\lambda}\right)^2} + \lambda \mathbf{I}}{2}. \tag{A.152}$$

$$\frac{|\lambda| \left(1 + \left(\frac{2\tilde{\mathbf{S}}}{\lambda}\right)^2 + O(\lambda^{-4})\right) + \lambda \mathbf{I}}{2} = \frac{|\lambda| + \lambda}{2} + \frac{\tilde{\mathbf{S}}^2}{|\lambda|} + O(\lambda^{-3}). \tag{A.153}$$

Hence

$$\lim_{\lambda \to \infty} \frac{\sqrt{\lambda^2 \mathbf{I} + 4\tilde{\mathbf{S}}^2} + \lambda \mathbf{I}}{2} = \lambda \mathbf{I}, \quad \lim_{\lambda \to -\infty} \frac{\sqrt{\lambda^2 \mathbf{I} + 4\tilde{\mathbf{S}}^2} + \lambda \mathbf{I}}{2} = \frac{\tilde{\mathbf{S}}^2}{|\lambda|} = -\frac{\tilde{\mathbf{S}}^2}{\lambda}. \tag{A.154}$$

Similarly,

$$\frac{\sqrt{\lambda^2 \mathbf{I} + 4\tilde{\mathbf{S}}^2} - \lambda \mathbf{I}}{2} = \frac{|\lambda| - \lambda}{2} + \frac{\tilde{\mathbf{S}}^2}{|\lambda|} + O(\lambda^{-3}), \tag{A.155}$$

$$\lim_{\lambda \to \infty} \frac{\sqrt{\lambda^2 \mathbf{I} + 4\tilde{\mathbf{S}}^2} - \lambda \mathbf{I}}{2} = \frac{\tilde{\mathbf{S}}^2}{\lambda}, \quad \lim_{\lambda \to -\infty} \frac{\sqrt{\lambda^2 \mathbf{I} + 4\tilde{\mathbf{S}}^2} - \lambda \mathbf{I}}{2} = \frac{\tilde{\mathbf{S}}^2}{|\lambda|} = -\lambda \mathbf{I}. \tag{A.156}$$

Since $\tilde{\mathbf{U}}, \tilde{\mathbf{V}}$ are independent of λ:

$$\lim_{\lambda \to \pm\infty} \mathbf{W}_2 \mathbf{W}_2^T = \tilde{\mathbf{U}} \left(\lim_{\lambda \to \pm\infty} \mathbf{S}_2 \right) \tilde{\mathbf{U}}^T. \tag{A.157}$$

$$\lim_{\lambda \to \pm\infty} \mathbf{W}_1^T \mathbf{W}_1 = \tilde{\mathbf{V}} \left(\lim_{\lambda \to \pm\infty} \mathbf{S}_1 \right) \tilde{\mathbf{V}}^T. \tag{A.158}$$

□

As $|\lambda| \to \infty$, one of the network representations approaches a scaled identity matrix, while the other tends toward zero. Intuitively, this suggests that the representations shift less and less as $|\lambda|$ increases. Next, we demonstrate that the NTK becomes progressively less variable as $|\lambda|$ grows and ultimately converges to zero.

A.3.2.4 NTK movement

Theorem A.3.6. *Under the assumptions of Theorem A.2.6, consider a linear network training on data $\mathbf{\Sigma}^{yx} = \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T$. At any arbitrary training time $t \geq 0$, let $\mathbf{W}_2(t)\mathbf{W}_1(t) = \mathbf{U}^*\mathbf{S}^*\mathbf{V}^{*T}$. Then,*

1. *For any $\lambda \in \mathbf{R}$:*

$$\begin{aligned} NTK(0) = \mathbf{I}_{N_o} \otimes \mathbf{X}^T\mathbf{V} &\begin{pmatrix} \frac{\sqrt{\lambda^2\mathbf{I}+4\mathbf{S}^{*2}}-\lambda\mathbf{I}}{2} & 0 \\ 0 & 0 \end{pmatrix} \mathbf{V}^T\mathbf{X} \\ + \mathbf{U} &\begin{pmatrix} \frac{\sqrt{\lambda^2\mathbf{I}+4\mathbf{S}^{*2}}+\lambda\mathbf{I}}{2} & 0 \\ 0 & 0 \end{pmatrix} \mathbf{U}^T \otimes \mathbf{X}^T\mathbf{X}. \end{aligned} \tag{A.159}$$

$$\begin{aligned} NTK(t) = \mathbf{I}_{N_o} \otimes \mathbf{X}^T\mathbf{V}^* &\begin{pmatrix} \frac{\sqrt{\lambda^2\mathbf{I}+4\mathbf{S}^{*2}}-\lambda\mathbf{I}}{2} & 0 \\ 0 & 0 \end{pmatrix} \mathbf{V}^{*T} \\ + \mathbf{U}^* &\begin{pmatrix} \frac{\sqrt{\lambda^2\mathbf{I}+4\mathbf{S}^{*2}}+\lambda\mathbf{I}}{2} & 0 \\ 0 & 0 \end{pmatrix} \mathbf{U}^{*T} \otimes \mathbf{X}^T\mathbf{X}. \end{aligned} \tag{A.160}$$

2. *As $\lambda \to \infty$:*

$$NTK(t) - NTK(0) \to \frac{1}{\lambda}\left(\mathbf{I}_{N_o} \otimes \mathbf{X}^T\mathbf{V}^*\tilde{\mathbf{S}}^{*2}\mathbf{V}^{*T}\mathbf{X} - \mathbf{I}_{N_o} \otimes \mathbf{X}^T\mathbf{V}\tilde{\mathbf{S}}^2\mathbf{V}^T\mathbf{X}\right) \to 0. \tag{A.161}$$

3. *As $\lambda \to -\infty$:*

$$NTK(t) - NTK(0) \to \frac{1}{\lambda}\left(\mathbf{U}\tilde{\mathbf{S}}^2\mathbf{U}^T \otimes \mathbf{X}^T\mathbf{X} - \mathbf{U}^*\tilde{\mathbf{S}}^{*2}\mathbf{U}^{*T} \otimes \mathbf{X}^T\mathbf{X}\right) \to 0. \tag{A.162}$$

Proof. These results follow from substituting the limiting forms of the network representations into the NTK definition (Eq. 3.6) and exploiting the linearity of the Kronecker product.

□

The theorem above demonstrates that as $|\lambda| \to \infty$, the NTK of a λ-Balanced network remains constant. This indicates that the network operates in the *lazy* regime throughout all training steps. The λ-balanced condition imposes a relationship between the singular values of the two weight matrices. Specifically, if $\mathbf{W}_2$ and $\mathbf{W}_1$ are λ-balanced and satisfy $\mathbf{W}_2\mathbf{W}_1 = \tilde{\mathbf{\Sigma}}_{yx}$, then for arbitrary singular values a_i, b_i, and s_i, the following relations hold:

$$a_i^2 - b_i^2 = \lambda, \quad a_i \cdot b_i = s_i.$$

As λ increases, the value of b_i must decrease. In the limit as $\lambda \to \infty$, $a_i^2 \to \lambda$ and $b_i^2 \to 0$. From the first equation, when $b_i^2 \to 0$, $a_i^2 \to \lambda$. Since these equations apply to all singular values of the matrices, it follows that for all i, $a_i^2 \to \lambda$, leading to the conclusion that:

$$\mathbf{W}_2^T\mathbf{W}_2 = \lambda\mathbf{I},$$

as expected. The intuition here is that the weights are constrained by the need to fit the data, which bounds their overall norms. The λ-balanced condition further specifies a relationship between these norms, and as $|\lambda|$ increases, this constraint tightens, driving $\mathbf{W}_2$ toward the identity matrix. In this regime, the network behaves similarly to a shallow network, with λ acting as a toggle between deep and shallow learning dynamics. This finding is significant as it highlights the impact of weight initialisation on learning regimes.

A.3.3 Representation robustness and sensitivity to noise

As derived in Braun et al. [40], the expected mean squared error under additive, independent and identically distributed input noise with mean $\mu = 0$ and variance

$\sigma_{\mathbf{x}}^2$ is

$$\left\langle \frac{1}{2P} \sum_{i=1}^{P} ||\mathbf{W}_2 \mathbf{W}_1 (\mathbf{x}_i + \xi_{\mathbf{x}}) - \mathbf{y}_i||_2^2 \right\rangle_{\xi_{\mathbf{x}}} = \sigma_{\mathbf{x}}^2 ||\mathbf{W}_2 \mathbf{W}_1||_F^2 + c, \tag{A.163}$$

where $c = \frac{1}{2}\operatorname{Tr}(\tilde{\mathbf{\Sigma}}^{yy}) - \frac{1}{2}\operatorname{Tr}(\tilde{\mathbf{\Sigma}}^{yx}\tilde{\mathbf{\Sigma}}^{yx^T})$ is a noise independent constant that only depends on the statistics of the training data. In Theorem A.3.3 we show that the network function converges to $\tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T$ and therefore

$$\sigma_{\mathbf{x}}^2 ||\mathbf{W}_2 \mathbf{W}_1||_F^2 = \sigma_{\mathbf{x}}^2 ||\tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T||_F^2 = \sigma_{\mathbf{x}}^2 ||\tilde{\mathbf{S}}||_F^2 = \sigma_{\mathbf{x}}^2 \sum_{i=1}^{N_h} \tilde{\mathbf{S}}_i^2. \tag{A.164}$$

As derived in Braun et al. [40], under the assumption of whitened inputs (Assumption 1), in the case of additive parameter noise with $\mu = 0$ and variance $\sigma_{\mathbf{W}}^2$, the expected mean squared error is

$$\begin{aligned} &\left\langle \frac{1}{2P} \sum_{i=1}^{P} ||(\mathbf{W}_2 + \xi_{\mathbf{W}_2})(\mathbf{W}_1 + \xi_{\mathbf{W}_1})\mathbf{x}_i - \mathbf{y}_i||_2^2 \right\rangle_{\xi_{\mathbf{W}_1}, \xi_{\mathbf{W}_2}} \\ &= \frac{1}{2} N_i \sigma_{\mathbf{W}}^2 ||\mathbf{W}_2||_F^2 + \frac{1}{2} N_o \sigma_{\mathbf{W}}^2 ||\mathbf{W}_1||_F^2 + \frac{1}{2} N_i N_h N_o \sigma^4 + c. \end{aligned} \tag{A.165}$$

Using Theorem A.3.3, we have

$$\begin{aligned} ||\mathbf{W}_1||_F^2 &= \operatorname{Tr}(\mathbf{W}_1^T \mathbf{W}_1) \\ &= \operatorname{Tr}\left(\frac{\sqrt{\lambda^2 \mathbf{I} + 4\tilde{\mathbf{S}}^2} + \lambda \mathbf{I}}{2} \right) \\ &= \frac{1}{2} \left(\sum_{i=1}^{N_h} \sqrt{\lambda^2 + 4\tilde{\mathbf{S}}_i^2} + \lambda \right) \end{aligned} \tag{A.166}$$

and

$$\begin{aligned} ||\mathbf{W}_2||_F^2 &= \operatorname{Tr}(\mathbf{W}_2 \mathbf{W}_2^T) \\ &= \operatorname{Tr}\left(\frac{\sqrt{\lambda^2 \mathbf{I} + 4\tilde{\mathbf{S}}^2} - \lambda \mathbf{I}}{2} \right) \end{aligned}$$

$$= \frac{1}{2}\left(\sum_{i=1}^{N_h} \sqrt{\lambda^2 + 4\tilde{\mathbf{S}}_i^2} - \lambda\right). \tag{A.167}$$

To find the λ that minimises the expected loss, we substitute the equations for the norms, take the partial derivative with respect to λ and set it to zero

$$\begin{aligned}
&\frac{\partial \langle \mathcal{L} \rangle_{\xi_{\mathbf{W}_1}, \xi_{\mathbf{W}_2}}}{\partial \lambda} \stackrel{!}{=} 0 \\
\Leftrightarrow &\frac{1}{4} N_i \sigma_{\mathbf{W}}^2 \frac{\partial}{\partial \lambda}\left(\sum_{i=1}^{N_h} \sqrt{\lambda^2 + 4\tilde{\mathbf{S}}_i^2} - \lambda\right) + \frac{1}{4} N_o \sigma_{\mathbf{W}}^2 \frac{\partial}{\partial \lambda}\left(\sum_{i=1}^{N_h} \sqrt{\lambda^2 + 4\tilde{\mathbf{S}}_i^2} + \lambda\right) = 0 \\
\Leftrightarrow &N_i \sum_{i=1}^{N_h} \frac{\lambda}{\sqrt{\lambda^2 + 4\tilde{\mathbf{S}}_i^2}} - N_i N_h + N_o \sum_{i=1}^{N_h} \frac{\lambda}{\sqrt{\lambda^2 + 4\tilde{\mathbf{S}}_i^2}} + N_o N_h = 0 \\
\Leftrightarrow &\sum_{i=1}^{N_h} \frac{\lambda}{\sqrt{\lambda^2 + 4\tilde{\mathbf{S}}_i^2}} = N_h \frac{N_i - N_o}{N_i + N_o}.
\end{aligned} \tag{A.168}$$

It follows, that under the assumption that $N_i = N_o$, the equation reduces to

$$\sum_{i=1}^{N_h} \frac{\lambda}{\sqrt{\lambda^2 + 4\tilde{\mathbf{S}}_i^2}} = 0. \tag{A.169}$$

We note, that the denominator is always positive and therefore, that the left-hand side of the equation is always larger zero for any $\lambda > 0$, and smaller than zero for any $\lambda < 0$. The euqation is therefore only solved for $\lambda = 0$.

A.3.4 Dynamics of the eigenvectors

A.3.4.1 Proof for Theorem 3.4.4

Let the input and output dimension of a two-layer linear network (Eq. 3.1) be equal, i.e., $N_i = N_o$, and $\lambda = 0$ then Eq. A.101 simplifies to

$$\begin{aligned}\mathbf{Q}\mathbf{Q}^T(t) = &\begin{bmatrix}\tilde{\mathbf{V}}\left(\mathbf{I}-e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{C}^T(\mathbf{B}^T)^{-1}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\right)\\ \tilde{\mathbf{U}}\left(\mathbf{I}+e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{C}^T(\mathbf{B}^T)^{-1}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\right)\end{bmatrix}\\ &\left[4e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{B}^{-1}\mathbf{S}^{-1}(\mathbf{B}^T)^{-1}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}+\left(\mathbf{I}-e^{-2\tilde{\mathbf{S}}\frac{t}{\tau}}\right)\tilde{\mathbf{S}}^{-1}\right.\\ &\left.-e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{B}^{-1}\mathbf{C}\left(e^{-2\tilde{\mathbf{S}}\frac{t}{\tau}}-\mathbf{I}\right)\tilde{\mathbf{S}}^{-1}\mathbf{C}^T(\mathbf{B}^T)^{-1}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\right]^{-1}\\ &\begin{bmatrix}\tilde{\mathbf{V}}\left(\mathbf{I}-e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{C}^T(\mathbf{B}^T)^{-1}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\right)\\ \tilde{\mathbf{U}}\left(\mathbf{I}+e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\mathbf{C}^T(\mathbf{B}^T)^{-1}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\right)\end{bmatrix}^T.\end{aligned} \tag{A.170}$$

See derivation in Sec. A.2.7.

Let the singular value decomposition of the input-output correlation of the task be

$$\mathrm{SVD}(\tilde{\mathbf{\Sigma}}^{yx}) = \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T, \tag{A.171}$$

and suppose that the initial state of the network can be written in the form

$$\mathrm{SVD}(\mathbf{W}_2(0)\mathbf{W}_1(0)) = \mathbf{U}\mathbf{S}\mathbf{V}^T = \tilde{\mathbf{U}}\mathbf{\Psi}(0)^T\mathbf{\Psi}(0)\tilde{\mathbf{V}}^T. \tag{A.172}$$

First, we note that the initial weights in this setting are not independent of the structure of the target task. In particular,

$$\begin{aligned}\mathbf{U}\sqrt{\mathbf{S}} &= \tilde{\mathbf{U}}\mathbf{\Psi}(0)^T\\ \Leftrightarrow \tilde{\mathbf{U}}^T\mathbf{U}\sqrt{\mathbf{S}} &= \mathbf{\Psi}(0)^T\\ \Leftrightarrow \sqrt{\mathbf{S}}\mathbf{U}^T\tilde{\mathbf{U}} &= \mathbf{\Psi}(0)\end{aligned} \tag{A.173}$$

and

$$\begin{aligned}\sqrt{\mathbf{S}}\mathbf{V}^T &= \mathbf{\Psi}(0)\tilde{\mathbf{V}}^T\\ \Leftrightarrow \sqrt{\mathbf{S}}\mathbf{V}^T\tilde{\mathbf{V}} &= \mathbf{\Psi}(0).\end{aligned} \tag{A.174}$$

and therefore

$$\begin{aligned}\sqrt{\mathbf{S}}\mathbf{U}^T\tilde{\mathbf{U}} &= \sqrt{\mathbf{S}}\mathbf{V}^T\tilde{\mathbf{V}}\\ \Leftrightarrow \mathbf{U}\mathbf{V}^T &= \tilde{\mathbf{U}}\tilde{\mathbf{V}}^T.\end{aligned} \tag{A.175}$$

This further simplifies the equation, as

$$\begin{aligned}\mathbf{U}\sqrt{\mathbf{S}} &= \tilde{\mathbf{U}}\boldsymbol{\Psi}(0)^T\\ \Leftrightarrow \mathbf{U} &= \tilde{\mathbf{U}}\boldsymbol{\Psi}(0)^T\sqrt{\mathbf{S}}^{-1},\end{aligned} \tag{A.176}$$

and

$$\begin{aligned}\sqrt{\mathbf{S}}\mathbf{V}^T &= \boldsymbol{\Psi}(0)\tilde{\mathbf{V}}^T\\ \Leftrightarrow \mathbf{V}^T &= \sqrt{\mathbf{S}}^{-1}\boldsymbol{\Psi}(0)\tilde{\mathbf{V}}^T\\ \Leftrightarrow \mathbf{V} &= \tilde{\mathbf{V}}\boldsymbol{\Psi}(0)^T\sqrt{\mathbf{S}}^{-1},\end{aligned} \tag{A.177}$$

then recollecting the definition of **B** and **C** we get

$$\begin{aligned}\mathbf{B}^T &= \tilde{\mathbf{U}}^T\mathbf{U} + \tilde{\mathbf{V}}^T\mathbf{V}\\ &= \tilde{\mathbf{U}}^T\tilde{\mathbf{U}}\boldsymbol{\Psi}(0)^T\sqrt{\mathbf{S}}^{-1} + \tilde{\mathbf{V}}^T\tilde{\mathbf{V}}\boldsymbol{\Psi}(0)^T\sqrt{\mathbf{S}}^{-1}\\ &= \left(\tilde{\mathbf{U}}^T\tilde{\mathbf{U}} + \tilde{\mathbf{V}}^T\tilde{\mathbf{V}}\right)\boldsymbol{\Psi}(0)^T\sqrt{\mathbf{S}}^{-1}\\ &= 2\boldsymbol{\Psi}(0)^T\sqrt{\mathbf{S}}^{-1},\end{aligned} \tag{A.178}$$

and

$$\begin{aligned}\mathbf{C}^T &= \tilde{\mathbf{U}}^T\mathbf{U} - \tilde{\mathbf{V}}^T\mathbf{V}\\ &= \left(\tilde{\mathbf{U}}^T\tilde{\mathbf{U}} - \tilde{\mathbf{V}}^T\tilde{\mathbf{V}}\right)\boldsymbol{\Psi}(0)^T\sqrt{\mathbf{S}}^{-1}\\ &= 0.\end{aligned} \tag{A.179}$$

Substituting the new values of **B** and **C** into Equation A.170 then yields

$$\mathbf{QQ}^T(t) =$$
$$\begin{bmatrix}\tilde{\mathbf{V}}\\ \tilde{\mathbf{U}}\end{bmatrix}\left[4e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\frac{1}{4}\mathbf{\Psi}(0)^{-1}\sqrt{\mathbf{S}}\mathbf{S}^{-1}\sqrt{\mathbf{S}}\mathbf{\Psi}(0)^{-T}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}+\left(\mathbf{I}-e^{-2\tilde{\mathbf{S}}\frac{t}{\tau}}\right)\tilde{\mathbf{S}}^{-1}\right]^{-1}\begin{bmatrix}\tilde{\mathbf{V}}\\ \tilde{\mathbf{U}}\end{bmatrix}^T$$
$$=\begin{bmatrix}\tilde{\mathbf{V}}\\ \tilde{\mathbf{U}}\end{bmatrix}\left[e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\left(\mathbf{\Psi}(0)^T\mathbf{\Psi}(0)\right)^{-1}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}+\left(\mathbf{I}-e^{-2\tilde{\mathbf{S}}\frac{t}{\tau}}\right)\tilde{\mathbf{S}}^{-1}\right]^{-1}\begin{bmatrix}\tilde{\mathbf{V}}\\ \tilde{\mathbf{U}}\end{bmatrix}^T. \qquad (A.180)$$

Finally, we note that the dynamics can thus be written as

$$\mathbf{QQ}^T(t) = \begin{bmatrix}\tilde{\mathbf{V}}\mathbf{\Psi}^T\mathbf{\Psi}(t)\tilde{\mathbf{V}}^T & \tilde{\mathbf{V}}\mathbf{\Psi}^T\mathbf{\Psi}(t)(t)\tilde{\mathbf{U}}^T\\ \tilde{\mathbf{U}}\mathbf{\Psi}^T\mathbf{\Psi}(t)\tilde{\mathbf{V}}^T & \tilde{\mathbf{U}}\mathbf{\Psi}^T\mathbf{\Psi}(t)\tilde{\mathbf{U}}^T\end{bmatrix} \qquad (A.181)$$

where

$$\mathbf{\Psi}^T\mathbf{\Psi}(t) = \left[e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}\left(\mathbf{\Psi}(0)^T\mathbf{\Psi}(0)\right)^{-1}e^{-\tilde{\mathbf{S}}\frac{t}{\tau}}+\left(\mathbf{I}-e^{-2\tilde{\mathbf{S}}\frac{t}{\tau}}\right)\tilde{\mathbf{S}}^{-1}\right]^{-1}. \qquad (A.182)$$

□

A.3.4.2 Solution for 2×2 dynamics

We consider small networks with input and output dimension $N_i = 2$ and $N_o = 2$. In this setting, the structure of the weight initialisation and task are encoded in the matrices

$$\mathbf{\Psi}(0)^T\mathbf{\Psi}(0) = \begin{bmatrix}\psi_1(0) & \nu(0)\\ \nu(0) & \psi_2(0)\end{bmatrix} \quad \text{and} \quad \tilde{\mathbf{S}} = \begin{bmatrix}\tilde{s}_1 & 0\\ 0 & \tilde{s}_2\end{bmatrix}, \qquad (A.183)$$

where the parameters $\psi_1(0)$ and $\psi_2(0)$ represent coupling within a singular mode, and $\nu(0)$ represents counterproductive cross-coupling between different singular modes at initialisation.

From Eq. 3.24, we have

$$\begin{aligned}
\mathbf{\Psi}^T\mathbf{\Psi}(t) = &\left[\begin{bmatrix} e^{\frac{-\tilde{s}_1 t}{\tau}} & 0 \\ 0 & e^{\frac{-\tilde{s}_2 t}{\tau}} \end{bmatrix} \begin{bmatrix} \psi_1(0) & \nu(0) \\ \nu(0) & \psi_2(0) \end{bmatrix}^{-1} \begin{bmatrix} e^{\frac{-\tilde{s}_1 t}{\tau}} & 0 \\ 0 & e^{\frac{-\tilde{s}_2 t}{\tau}} \end{bmatrix} \right. \\
&\left. + \left[\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} e^{\frac{-2\tilde{s}_1 t}{\tau}} & 0 \\ 0 & e^{\frac{-2\tilde{s}_2 t}{\tau}} \end{bmatrix} \right] \begin{bmatrix} \tilde{s}_1 & 0 \\ 0 & \tilde{s}_2 \end{bmatrix}^{-1} \right]^{-1}
\end{aligned}$$

$$\begin{aligned}
= &\left[\frac{1}{\psi_1(0)\psi_2(0) - \nu(0)^2} \begin{bmatrix} e^{\frac{-\tilde{s}_1 t}{\tau}} & 0 \\ 0 & e^{\frac{-\tilde{s}_2 t}{\tau}} \end{bmatrix} \begin{bmatrix} \psi_2(0) & -\nu(0) \\ -\nu(0) & \psi_1(0) \end{bmatrix} \begin{bmatrix} e^{\frac{-\tilde{s}_1 t}{\tau}} & 0 \\ 0 & e^{\frac{-\tilde{s}_2 t}{\tau}} \end{bmatrix} \right. \\
+ &\left. \left[\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} - \begin{bmatrix} e^{\frac{-2\tilde{s}_1 t}{\tau}} & 0 \\ 0 & e^{\frac{-2\tilde{s}_2 t}{\tau}} \end{bmatrix} \right] \begin{bmatrix} \frac{1}{\tilde{s}_1} & 0 \\ 0 & \frac{1}{\tilde{s}_2} \end{bmatrix} \right]^{-1}, \qquad \text{(A.184)}
\end{aligned}$$

where we use

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}. \qquad \text{(A.185)}$$

We continue with

$$\begin{aligned}
\mathbf{\Psi}^T\mathbf{\Psi}(t) = &\left[\frac{1}{\psi_1(0)\psi_2(0) - \nu(0)^2} \begin{bmatrix} e^{\frac{-\tilde{s}_1 t}{\tau}} & 0 \\ 0 & e^{\frac{-\tilde{s}_2 t}{\tau}} \end{bmatrix} \begin{bmatrix} \psi_2(0) & -\nu(0) \\ -\nu(0) & \psi_1(0) \end{bmatrix} \begin{bmatrix} e^{\frac{-\tilde{s}_1 t}{\tau}} & 0 \\ 0 & e^{\frac{-\tilde{s}_2 t}{\tau}} \end{bmatrix} \right. \\
&\left. + \begin{bmatrix} \frac{1}{\tilde{s}_1} & 0 \\ 0 & \frac{1}{\tilde{s}_2} \end{bmatrix} - \begin{bmatrix} \frac{1}{\tilde{s}_1} e^{\frac{-2\tilde{s}_1 t}{\tau}} & 0 \\ 0 & \frac{1}{\tilde{s}_2} e^{\frac{-2\tilde{s}_2 t}{\tau}} \end{bmatrix} \right]^{-1}
\end{aligned}$$

$$\begin{aligned}
= &\left[\frac{1}{\psi_1(0)\psi_2(0) - \nu(0)^2} \begin{bmatrix} e^{\frac{-2\tilde{s}_1 t}{\tau}} \psi_2(0) & -e^{\frac{-\tilde{s}_1 t}{\tau}} \nu(0) e^{\frac{-\tilde{s}_2 t}{\tau}} \\ -e^{\frac{-\tilde{s}_2 t}{\tau}} \nu(0) e^{\frac{-\tilde{s}_1 t}{\tau}} & e^{\frac{-2\tilde{s}_2 t}{\tau}} \psi_1(0) \end{bmatrix} \right. \\
&\left. + \begin{bmatrix} \frac{1}{\tilde{s}_1} & 0 \\ 0 & \frac{1}{\tilde{s}_2} \end{bmatrix} - \begin{bmatrix} \frac{1}{\tilde{s}_1} e^{\frac{-2\tilde{s}_1 t}{\tau}} & 0 \\ 0 & \frac{1}{\tilde{s}_2} e^{\frac{-2\tilde{s}_2 t}{\tau}} \end{bmatrix} \right]^{-1} \\
= &\begin{bmatrix} \frac{e^{\frac{-2\tilde{s}_1 t}{\tau}} \psi_2(0)}{\psi_1(0)\psi_2(0) - \nu(0)^2} + \frac{1}{\tilde{s}_1} - \frac{1}{\tilde{s}_1} e^{\frac{-2\tilde{s}_1 t}{\tau}} & -\frac{e^{\frac{-\tilde{s}_1 t}{\tau}} \nu(0) e^{\frac{-\tilde{s}_2 t}{\tau}}}{\psi_1(0)\psi_2(0) - \nu(0)^2} \\ -\frac{e^{\frac{-\tilde{s}_2 t}{\tau}} \nu(0) e^{\frac{-\tilde{s}_1 t}{\tau}}}{\psi_1(0)\psi_2(0) - \nu(0)^2} & \frac{e^{\frac{-2\tilde{s}_2 t}{\tau}} \psi_1(0)}{\psi_1(0)\psi_2(0) - \nu(0)^2} + \frac{1}{\tilde{s}_2} - \frac{1}{\tilde{s}_2} e^{\frac{-2\tilde{s}_2 t}{\tau}} \end{bmatrix}^{-1}. \qquad \text{(A.186)}
\end{aligned}$$

We use Eq. A.185 and simplify the denominator

$$\mathbf{\Psi}^T\mathbf{\Psi}(t) = \frac{1}{\left(\frac{e^{\frac{-2\tilde{s}_2 t}{\tau}}\psi_1(0)}{\psi_1(0)\psi_2(0)-\nu(0)^2}+\frac{1}{\tilde{s}_2}-\frac{1}{\tilde{s}_2}e^{\frac{-2\tilde{s}_2 t}{\tau}}\right)\left(\frac{e^{\frac{-2\tilde{s}_1 t}{\tau}}\psi_2(0)}{\psi_1(0)\psi_2(0)-\nu(0)^2}+\frac{1}{\tilde{s}_1}-\frac{1}{\tilde{s}_1}e^{\frac{-2\tilde{s}_1 t}{\tau}}\right)-\left(-\frac{e^{\frac{-\tilde{s}_2 t}{\tau}}\nu(0)e^{\frac{-\tilde{s}_1 t}{\tau}}}{\psi_1(0)\psi_2(0)-\nu(0)^2}\right)^2} \tag{A.187}$$

$$\times\begin{bmatrix} \frac{e^{\frac{-2\tilde{s}_2 t}{\tau}}\psi_1(0)}{\psi_1(0)\psi_2(0)-\nu(0)^2}+\frac{1}{\tilde{s}_2}-\frac{1}{\tilde{s}_2}e^{\frac{-2\tilde{s}_2 t}{\tau}} & \frac{e^{\frac{-\tilde{s}_1 t}{\tau}}\nu(0)e^{\frac{-\tilde{s}_2 t}{\tau}}}{\psi_1(0)\psi_2(0)-\nu(0)^2} \\ \frac{e^{\frac{-\tilde{s}_2 t}{\tau}}\nu(0)e^{\frac{-\tilde{s}_1 t}{\tau}}}{\psi_1(0)\psi_2(0)-\nu(0)^2} & \frac{e^{\frac{-2\tilde{s}_1 t}{\tau}}\psi_2(0)}{\psi_1(0)\psi_2(0)-\nu(0)^2}+\frac{1}{\tilde{s}_1}-\frac{1}{\tilde{s}_1}e^{\frac{-2\tilde{s}_1 t}{\tau}} \end{bmatrix}.$$

The diagonal element $\psi_1(t)$ is given as

$$\psi_1(t) = \frac{\frac{e^{\frac{-2\tilde{s}_2 t}{\tau}}\psi_1(0)}{\psi_1(0)\psi_2(0)-\nu(0)^2}+\frac{1}{\tilde{s}_2}-\frac{1}{\tilde{s}_2}e^{\frac{-2\tilde{s}_2 t}{\tau}}}{D_{num}}, \tag{A.188}$$

where

$$\begin{aligned} D_{num} =& \Big(\frac{e^{\frac{-2\tilde{s}_2 t}{\tau}}\psi_1(0)}{\psi_1(0)\psi_2(0)-\nu(0)^2}+\frac{1}{\tilde{s}_2}-\frac{1}{\tilde{s}_2}e^{\frac{-2\tilde{s}_2 t}{\tau}}\Big)\Big(\frac{e^{\frac{-2\tilde{s}_1 t}{\tau}}\psi_2(0)}{\psi_1(0)\psi_2(0)-\nu(0)^2} \\ &+\frac{1}{\tilde{s}_1}-\frac{1}{\tilde{s}_1}e^{\frac{-2\tilde{s}_1 t}{\tau}}\Big)-\Big(-\frac{e^{\frac{-\tilde{s}_2 t}{\tau}}\nu(0)e^{\frac{-\tilde{s}_1 t}{\tau}}}{\psi_1(0)\psi_2(0)-\nu(0)^2}\Big)^2 \end{aligned} \tag{A.189}$$

and interchanging subscripts 1 and 2 yields $\psi_2(t)$. As a check on this result, by setting $\nu(0)=0$ we recover the expression

$$\psi_1(t) = \frac{\psi_1(0)}{e^{\frac{-2\tilde{s}_1 t}{\tau}}+\frac{\psi_1(0)}{\tilde{s}_1}\left(1-e^{\frac{-2\tilde{s}_1 t}{\tau}}\right)}, \tag{A.190}$$

from Saxe et al. [247]. We further simplify the denominator to

$$\mathbf{\Psi}^T\mathbf{\Psi}(t) = \frac{1}{\frac{1}{\psi_1(0)\psi_2(0)-\nu(0)^2}\left(e^{\frac{-2(\tilde{s}_1+\tilde{s}_2)t}{\tau}}\left(1-\frac{\psi_1(0)}{\tilde{s}_1}-\frac{\psi_2(0)}{\tilde{s}_2}\right)+e^{\frac{-2\tilde{s}_2 t}{\tau}}\frac{\psi_1(0)}{\tilde{s}_1}+e^{\frac{-2\tilde{s}_1 t}{\tau}}\frac{\psi_2(0)}{\tilde{s}_2}\right)+\frac{1}{\tilde{s}_2\tilde{s}_1}} \tag{A.191}$$

$$\times\begin{bmatrix} \frac{e^{\frac{-2\tilde{s}_2 t}{\tau}}\psi_1(0)}{\psi_1(0)\psi_2(0)-\nu(0)^2}+\frac{1}{\tilde{s}_2}-\frac{1}{\tilde{s}_2}e^{\frac{-2\tilde{s}_2 t}{\tau}} & \frac{e^{\frac{-\tilde{s}_1 t}{\tau}}\nu(0)e^{\frac{-\tilde{s}_2 t}{\tau}}}{\psi_1(0)\psi_2(0)-\nu(0)^2} \\ \frac{e^{\frac{-\tilde{s}_2 t}{\tau}}\nu(0)e^{\frac{-\tilde{s}_1 t}{\tau}}}{\psi_1(0)\psi_2(0)-\nu(0)^2} & \frac{e^{\frac{-2\tilde{s}_1 t}{\tau}}\psi_2(0)}{\psi_1(0)\psi_2(0)-\nu(0)^2}+\frac{1}{\tilde{s}_1}-\frac{1}{\tilde{s}_1}e^{\frac{-2\tilde{s}_1 t}{\tau}} \end{bmatrix}$$

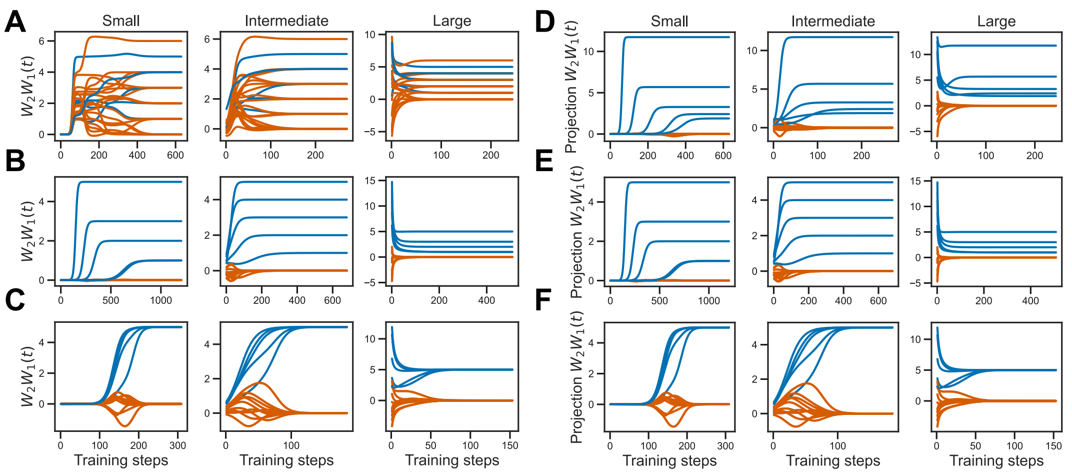


Figure A.2: A-C Network function dynamics (Diagonal elements: blue, Off-diagonal elements: red) learning with learning rate $\eta = 0.01$ on the target 5×5 diagonal matrices shown in Eq. A.194. The network was initialised as defined in Sec.A.3.4 with Small ($\sigma = 1e{-}6$), Intermediate ($\sigma = 0.1$) and Large ($\sigma = 2$) variance, and hidden layer size $N_h = 10$. **A**, Dense. **B**, Diagonal. **C**, Equal diagonal. **D-F**. Corresponding numerical temporal dynamics of the projection of the network function on- and off-diagonal elements into the singular-basis of the initialisation. Equivalently, the temporal dynamics of the elements of $\mathbf{\Psi}\mathbf{\Psi}^T$ bottom left quadrant. **D**, Dense. **E**, Diagonal. **F**, Equal diagonal.

A.3.4.3 Off-Diagonal decoupling dynamics

We track the decoupling by considering the dynamics of the off-diagonal element.

$$\nu(t) = \frac{\frac{e^{\frac{-\tilde{s}_2 t}{\tau}}\nu(0)e^{\frac{-\tilde{s}_1 t}{\tau}}}{\psi_1(0)\psi_2(0)-\nu(0)^2}}{\frac{1}{\psi_1(0)\psi_2(0)-\nu(0)^2}\left(e^{\frac{-2(\tilde{s}_1+\tilde{s}_2)t}{\tau}}\left(1-\frac{\psi_1(0)}{\tilde{s}_1}-\frac{\psi_2(0)}{\tilde{s}_2}\right)+e^{\frac{-2\tilde{s}_2 t}{\tau}}\frac{\psi_1(0)}{\tilde{s}_1}+e^{\frac{-2\tilde{s}_1 t}{\tau}}\frac{\psi_2(0)}{\tilde{s}_2}\right)+\frac{1}{\tilde{s}_2\tilde{s}_1}}. \tag{A.192}$$

As t tends to infinity $\lim_{t\to\infty}\nu(t) = 0$ the off-diagonal element shrinks to zero. We can further simplify the off-diagonal to

$$\nu(t) = \frac{\nu(0)}{e^{\frac{-(\tilde{s}_1+\tilde{s}_2)t}{\tau}}\left(1-\frac{\psi_1(0)}{\tilde{s}_1}-\frac{\psi_2(0)}{\tilde{s}_2}\right)+e^{\frac{(\tilde{s}_1-\tilde{s}_2)t}{\tau}}\frac{\psi_1(0)}{\tilde{s}_1}+e^{\frac{(\tilde{s}_2-\tilde{s}_1)t}{\tau}}\frac{\psi_2(0)}{\tilde{s}_2}+\frac{\psi_1(0)\psi_2(0)-\nu(0)^2}{\tilde{s}_2\tilde{s}_1}}. \tag{A.193}$$

Eq. A.193 can exhibit non-monotonic trajectories with transient peaks as shown in Fig. 3.5. The qualitative observations for the 2×2 network hold for larger target matrices as shown in Fig. A.2. For large initialisation, the dynamics are exponential.

At intermediate and small initialisation, the maximum of the off-diagonal is reached before the singular mode is fully learned. Generally, at intermediate variance initialisations, we observe more complex behaviour. In the small initialisation scheme, the peak is of negligible size. The respective target matrix for Panel A-D, B-E and C-F in Fig. A.2 are

$$\text{Dense:} \begin{bmatrix} 5 & 6 & 3 & 0 & 1 \\ 4 & 1 & 0 & 1 & 2 \\ 3 & 0 & 2 & 4 & 0 \\ 3 & 4 & 0 & 3 & 2 \\ 2 & 0 & 1 & 3 & 4 \end{bmatrix}, \tag{A.194}$$

$$\text{Diagonal:} \begin{bmatrix} 5 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 3 & 0 \\ 0 & 0 & 0 & 0 & 4 \end{bmatrix}, \tag{A.195}$$

$$\text{Equal Diagonal:} \begin{bmatrix} 5 & 0 & 0 & 0 & 0 \\ 0 & 5 & 0 & 0 & 0 \\ 0 & 0 & 5 & 0 & 0 \\ 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 5 \end{bmatrix}. \tag{A.196}$$

We characterise these dynamics considering the case where $\tilde{s}_1 = \tilde{s}_2 = \tilde{s}$ for the two-by-two solution (i.e. equal diagonal target $\mathbf{y}$) for which we can compute the time of the peak. In this particular case, we can further simplify the off-diagonal to

$$\nu(t) = \frac{\nu(0)}{e^{\frac{-2(\tilde{s})t}{\tau}}\left(1 - \frac{\psi_1(0)+\psi_2(0)}{\tilde{s}}\right) + \frac{\psi_1(0)+\psi_2(0)}{\tilde{s}} + \frac{\psi_1(0)\psi_2(0)-\nu(0)^2}{\tilde{s}^2}}. \tag{A.197}$$

We find the time of the maximum of the off-diagonal elements to be $t_{peak} = \frac{\tau}{4\tilde{s}} \ln \frac{\tilde{s}(\tilde{s}-\psi_1(0)-\psi_2(0))}{\psi_1(0)\psi_2(0)-\nu(0)^2}$.

The presence of a peak in the off-diagonal values, indicates the decoupling, but as shown in Fig. 3.5D-F, the peak size is negligible in comparison to the size of the on-diagonal values for small initial weights. This difference is reminiscent of the silent alignment effect described by Atanasov et al. [18]. We further note, that the time scale of decoupling is on the same order as the one reported for the silent alignment effect $t_{sa} = \frac{1}{\tilde{s}}$.

A.3.4.4 On-diagonal dynamics and the effect of initialisation variance

In this section we revisit the impact of initialisation scale for the on-diagonal dynamics. We start from Eq. A.191

$$\psi_1(t) = \tag{A.198}$$
$$\frac{\frac{e^{\frac{-2\tilde{s}_2 t}{\tau}}\psi_1(0)}{\psi_1(0)\psi_2(0)-\nu(0)^2} + \frac{1}{\tilde{s}_2} - \frac{1}{\tilde{s}_2}e^{\frac{-2\tilde{s}_2 t}{\tau}}}{\frac{1}{\psi_1(0)\psi_2(0)-\nu(0)^2}\left(e^{\frac{-2(\tilde{s}_1+\tilde{s}_2)t}{\tau}}\left(1-\frac{\psi_1(0)}{\tilde{s}_1}-\frac{\psi_2(0)}{\tilde{s}_2}\right)+e^{\frac{-2\tilde{s}_2 t}{\tau}}\frac{\psi_1(0)}{\tilde{s}_1}+e^{\frac{-2\tilde{s}_1 t}{\tau}}\frac{\psi_2(0)}{\tilde{s}_2}\right)+\frac{1}{\tilde{s}_2\tilde{s}_1}}.$$

The diagonal elements simplify in the cases where $\tilde{s}_1 = \tilde{s}_2 = \tilde{s}$ (i.e. target $\mathbf{Y}$ is diagonal),

$$\psi_1(t) = \tag{A.199}$$
$$\frac{\frac{e^{\frac{-2\tilde{s}t}{\tau}}\psi_1(0)}{\psi_1(0)\psi_2(0)-\nu(0)^2} + \frac{1}{\tilde{s}} - \frac{1}{\tilde{s}}e^{\frac{-2\tilde{s}t}{\tau}}}{\frac{1}{\psi_1(0)\psi_2(0)-\nu(0)^2}\left(e^{\frac{-4\tilde{s}t}{\tau}}\left(1-\frac{\psi_1(0)}{\tilde{s}}-\frac{\psi_2(0)}{\tilde{s}}\right)+e^{\frac{-2\tilde{s}t}{\tau}}\frac{\psi_1(0)}{\tilde{s}}+e^{\frac{-2\tilde{s}t}{\tau}}\frac{\psi_2(0)}{\tilde{s}}\right)+\frac{1}{\tilde{s}^2}}.$$

We consider when $|\psi_1(0)|, |\psi_2(0)|, |\nu(0)| \ll 1$, and recover a sigmoidal trajectory,

$$\psi_1(t) = \frac{\tilde{s}\psi_1(0)}{e^{\frac{-2\tilde{s}t}{\tau}}\left[\tilde{s}-\psi_1(0)-\psi_2(0)\right]+\psi_1(0)+\psi_2(0)}. \tag{A.200}$$

We can compute the time at which $\psi_1(t)$ rises to half its asymptotic value to be

$$t_{\text{half}} = \frac{\tau}{2\tilde{s}}\log\left(\frac{\tilde{s}-\psi_1(0)-\psi_2(0)}{\psi_1(0)-\psi_2(0)}\right). \tag{A.201}$$

For $|\psi_1(0)|, |\psi_2(0)|, |\nu(0)| \gg 0$ the dynamics of the on-diagonal element ψ_1 is close to exponential.

A.3.5 The impact of the architecture

A.3.5.1 Overview of Mirror Flow Analysis for Implicit Bias

Here we recap the standard analysis for determining the implicit bias of a linear network through mirror flow. As first introduced in Gunasekar et al. [111], if the learning dynamics of the predictor $\boldsymbol{\beta}$ can be expressed as a *mirror flow* for some strictly convex potential $\Phi_\alpha(\boldsymbol{\beta})$,

$$\dot{\boldsymbol{\beta}} = -\left(\nabla^2\Phi_\alpha(\boldsymbol{\beta})\right)^{-1}\boldsymbol{X}\boldsymbol{\rho}, \tag{A.202}$$

where $\boldsymbol{\rho} = (\boldsymbol{X}^\intercal\boldsymbol{\beta} - \boldsymbol{y})$ is the residual, then the limiting solution of the dynamics is determined by the constrained optimisation problem,

$$\boldsymbol{\beta}(\infty) = \underset{\boldsymbol{\beta}\in\mathbb{R}^{N_i}}{\arg\min} D_\Phi(\boldsymbol{\beta},\boldsymbol{\beta}(0)) \quad \text{s.t.} \quad \boldsymbol{X}^\intercal\boldsymbol{\beta} = \boldsymbol{y}, \tag{A.203}$$

where $D_\Phi(\boldsymbol{p},\boldsymbol{q}) = \Phi(\boldsymbol{p}) - \Phi(\boldsymbol{q}) - \langle\nabla\Phi(\boldsymbol{q}),\boldsymbol{p}-\boldsymbol{q}\rangle$ is the Bregman divergence defined with Φ. In this analysis we consider $N_o = 0$

To understand the relationship between mirror flow eq. (A.202) and the optimisation problem eq. (A.203), we consider an equivalent constrained optimisation problem

$$\boldsymbol{\beta}(\infty) = \underset{\boldsymbol{\beta}\in\mathbb{R}^{N_i}}{\arg\min}\, Q(\boldsymbol{\beta}) \quad \text{s.t.} \quad \boldsymbol{X}\boldsymbol{\beta} = \boldsymbol{y}, \tag{A.204}$$

where $Q(\boldsymbol{\beta}) = \Phi_\alpha(\boldsymbol{\beta}) - \nabla\Phi_\alpha(\boldsymbol{\beta}(0))^\intercal\boldsymbol{\beta}$, which is often referred to as the *implicit bias*. $Q(\boldsymbol{\beta})$ is strictly convex, and thus it is sufficient to show that $\boldsymbol{\beta}(\infty)$ is a first order KKT point of the constrained optimisation (A.204). This is true iff there exists $\boldsymbol{\nu} \in \mathbb{R}^n$ such that $\nabla Q(\boldsymbol{\beta}(\infty)) = \boldsymbol{X}^\intercal\boldsymbol{\nu}$. The goal is to derive $\boldsymbol{\nu}$ from the mirror flow eq. (A.202). Notice, we can rewrite eq. (A.202) as, $(\nabla\dot{\Phi_\alpha}(\boldsymbol{\beta})) = -\boldsymbol{X}^\intercal\boldsymbol{\rho}$, which integrated over time gives

$$\nabla\Phi_\alpha(\boldsymbol{\beta}(\infty)) - \nabla\Phi_\alpha(\boldsymbol{\beta}(0)) = -\boldsymbol{X}^\intercal\int_0^\infty \boldsymbol{\rho}(t)dt. \tag{A.205}$$

The LHS is $\nabla Q(\boldsymbol{\beta}(\infty))$. Thus, by defining $\boldsymbol{\nu} = \int_0^\infty \boldsymbol{\rho}(t)dt$, which assumes the residual decays fast enough such that this is well defined, then we have shown the desired KKT condition. Crucial to this analysis is that there exists a solution to the second-order differential equation

$$\nabla^2 \Phi_\alpha(\boldsymbol{\beta}) = \left(\nabla_\theta \boldsymbol{\beta} \nabla_\theta \boldsymbol{\beta}^\intercal\right)^{-1}, \tag{A.206}$$

which even for extremely simple Jacobian maps may not be true [113].

A.3.5.2 Deriving M

We consider the dynamics of a two-layer linear network with h hidden neurons and c outputs, $f(x;\theta) = \mathbf{W}_2\mathbf{W}_1\mathbf{X}$, where $\mathbf{W}_1 \in \mathbb{R}^{N_h \times N_i}$ and $\mathbf{W}_2 \in \mathbb{R}^{N_o \times N_h}$. We assume that $N_h \geq \min(N_i, N_o)$, such that this parameterisation can represent all linear maps from $\mathbb{R}_i^N \to \mathbb{R}^{N_o}$. The rescaling symmetry in this model between the first and second layer implies the $N_h \times N_h$ matrix $\mathbf{\Lambda} = \mathbf{W}_2^\intercal \mathbf{W}_2 - \mathbf{W}_1 \mathbf{W}_1^\intercal$ determined at initialisation remains conserved throughout gradient flow [74]. This can be easily shown from the temporal dynamics of A and W,

$$\dot{\mathbf{W}}_2{}^\intercal = -\eta_{w_2} \mathbf{W}_1 \mathbf{X}(\mathbf{X}^\intercal \boldsymbol{\beta} - \mathbf{Y}^\intercal), \tag{A.207}$$

$$\dot{\mathbf{W}}_1{}^\intercal = -\eta_{w_1} \mathbf{X}(\mathbf{X}^\intercal \boldsymbol{\beta} - \mathbf{Y}^\intercal)\mathbf{W}_2. \tag{A.208}$$

The NTK matrix can be expressed as

$$\mathbf{NTK} = (\mathbf{I}_{N_o} \otimes \mathbf{X}^\intercal)\left(\eta_{w_1} \mathbf{W}_2 \mathbf{W}_2^\intercal \oplus \eta_{w_2} \mathbf{W}_1^\intercal \mathbf{W}_1\right)(\mathbf{I}_{N_o} \otimes \mathbf{X}), \tag{A.209}$$

where $\otimes$ and $\oplus$ denote the Kronecker product and sum respectively. The Kronecker sum is defined for square matrices $\mathbf{C} \in \mathbb{R}^{c \times c}$ and $\mathbf{D} \in \mathbb{R}^{d \times d}$ as $\mathbf{C} \oplus \mathbf{D} = \mathbf{C} \otimes \mathbf{I}_d + \mathbf{I}_c \otimes \mathbf{D}$.

We consider the dynamics of $\boldsymbol{\beta} = \mathbf{W}_1^\intercal \mathbf{W}_2^\intercal \in \mathbb{R}^{N_i \times N_o}$ in function space, which is governed by the ODE,

$$\dot{\boldsymbol{\beta}} = \mathbf{W}_1^\intercal \dot{\mathbf{W}}_2{}^\intercal + \dot{\mathbf{W}}_1{}^\intercal \mathbf{W}_2^\intercal = -\left(\eta_{w_2}\mathbf{W}_1^\intercal \mathbf{W}_1 \mathbf{X}(\mathbf{X}^\intercal \boldsymbol{\beta} - \mathbf{Y}^\intercal) + \eta_{w_1}\mathbf{X}(\mathbf{X}^\intercal\boldsymbol{\beta} - \mathbf{Y}^\intercal)\mathbf{W}_2\mathbf{W}_2^\intercal.\right) \tag{A.210}$$

Vectorising using the identity $\mathrm{vec}(\mathbf{ABC}) = (\mathbf{C}^\intercal \otimes \mathbf{A})\mathrm{vec}(\mathbf{B})$ Eq. A.210 becomes

$$\begin{aligned}
\mathrm{vec}\left(\dot{\boldsymbol{\beta}}\right) &= -\mathrm{vec}\left(\eta_{w_1}\mathbf{I}_{N_i}\mathbf{X}(\mathbf{X}^\intercal\boldsymbol{\beta} - \mathbf{Y}^\intercal)\mathbf{W}_2\mathbf{W}_2^\intercal + \eta_{w_2}\mathbf{W}_1^\intercal\mathbf{W}_1\mathbf{X}(\mathbf{X}^\intercal\boldsymbol{\beta} - \mathbf{Y}^\intercal)\mathbf{I}_{N_o}\right), \\
&= -(\eta_{w_1}\mathbf{W}_2\mathbf{W}_2^\intercal \otimes \mathbf{I}_{N_i} + \eta_{w_2}\mathbf{I}_{N_o} \otimes \mathbf{W}_1^\intercal\mathbf{W}_1)\mathrm{vec}(\mathbf{X}\mathbf{X}^\intercal\boldsymbol{\beta} - \mathbf{X}\mathbf{Y}^\intercal)), \\
&= -\underbrace{\left(\eta_{w_1}\mathbf{W}_2\mathbf{W}_2^\intercal \oplus \eta_{w_2}\mathbf{W}_1^\intercal\mathbf{W}_1\right)}_{\mathbf{M}}\mathrm{vec}(\mathbf{X}\mathbf{X}^\intercal\boldsymbol{\beta} - \mathbf{X}\mathbf{Y}^\intercal).
\end{aligned} \tag{A.211}$$

We find that the dynamics of $\boldsymbol{\beta}$ can be expressed as gradient flow preconditioned by a matrix $\mathbf{M}$ that depends on quadratics of $\mathbf{W}_2$ and $\mathbf{W}_1$.

Theorem A.3.7. *Whenever $||\boldsymbol{\beta}_k||_F \neq 0$ for all $k \in [N_h]$, the matrix $\mathbf{M}$ can be expressed as the sum $\mathbf{M} = \sum_{k=1}^{h}\mathbf{M}_k$ over hidden neurons where M_k is defined as,*

$$\mathbf{M}_k = \left(\frac{\sqrt{\lambda_k^2 + 4\eta_{w_1}\eta_{w_2}||\boldsymbol{\beta}_k||_F^2} + \lambda_k}{2}\right)\frac{\boldsymbol{\beta}_k^\intercal\boldsymbol{\beta}_k}{||\boldsymbol{\beta}_k||_F^2} \oplus \left(\frac{\sqrt{\lambda_k^2 + 4\eta_{w_1}\eta_{w_2}||\boldsymbol{\beta}_k||_F^2} - \lambda_k}{2}\right)\frac{\boldsymbol{\beta}_k\boldsymbol{\beta}_k^\intercal}{||\boldsymbol{\beta}_k||_F^2}. \tag{A.212}$$

Proof. Consider a single hidden neuron $k \in [N_h]$ of the multi-output model defined by the parameters $\mathbf{W}_{1k} \in \mathbb{R}^{N_i}$ and $\mathbf{W}_{2k} \in \mathbb{R}^{N_o}$. Let $\boldsymbol{\beta}_k = \mathbf{W}_{1k}^\intercal\mathbf{W}_{2k}^\intercal$ be the $\mathbb{R}^{N_i \times N_o}$ matrix representing the contribution of this hidden neuron to the input-output map of the network $\boldsymbol{\beta} = \sum_{k=1}^{Nh}\boldsymbol{\beta}_k$. Consider the two gram matrices $\boldsymbol{\beta}_k^\intercal\boldsymbol{\beta}_k \in \mathbb{R}^{N_o \times N_o}$ and $\boldsymbol{\beta}_k\boldsymbol{\beta}_k^\intercal \in \mathbb{R}^{N_i \times N_i}$,

$$\boldsymbol{\beta}_k^\intercal\boldsymbol{\beta}_k = ||\mathbf{W}_{1k}||^2\mathbf{W}_{2k}\mathbf{W}_{2k}^\intercal, \qquad \boldsymbol{\beta}_k\boldsymbol{\beta}_k^\intercal = ||\mathbf{W}_{2k}||^2\mathbf{W}_{1k}^\intercal\mathbf{W}_{1k}. \tag{A.213}$$

Notice that we can express $||\boldsymbol{\beta}_k||_F^2$ as

$$||\boldsymbol{\beta}_k||_F^2 = \mathrm{Tr}(\boldsymbol{\beta}_k^\intercal\boldsymbol{\beta}_k) = \mathrm{Tr}(\boldsymbol{\beta}_k\boldsymbol{\beta}_k^\intercal) = ||\mathbf{W}_{2k}||^2||\mathbf{W}_{1k}||^2 \tag{A.214}$$

At each hidden neuron we have the conserved quantity[1] $\eta_{w_1}||\mathbf{W}_{2k}||^2 - \eta_{w_2}||\mathbf{W}_{1k}||^2 = \lambda_k$ where $\lambda_k \in \mathbb{R}$. Using this quantity we can invert the expression for $||\boldsymbol{\beta}_k||_F^2$ to get

$$||\mathbf{W}_{2k}||^2 = \frac{\sqrt{\lambda_k^2 + 4\eta_{w_2}\eta_{w_1}||\boldsymbol{\beta}_k||_F^2} + \lambda_k}{2\eta_{w_1}}, \tag{A.215}$$

$$||\mathbf{W}_{1k}||^2 = \frac{\sqrt{\lambda_k^2 + 4\eta_{w_2}\eta_{w_1}||\boldsymbol{\beta}_k||_F^2} - \lambda_k}{2\eta_{w_2}}. \tag{A.216}$$

When $||\boldsymbol{\beta}_k||_F^2 > 0$, we can use these expressions to solve for the outer products $\mathbf{W}_{2k}\mathbf{W}_{2k}^\intercal$ and $\mathbf{W}_{1k}^\intercal\mathbf{W}_{1k}$ in terms of $\boldsymbol{\beta}_k$ and λ_k,

$$\mathbf{W}_{2k}\mathbf{W}_{2k}^\intercal = \frac{\sqrt{\lambda_k^2 + 4\eta_{w_2}\eta_{w_1}||\boldsymbol{\beta}_k||_F^2} + \lambda_k}{2\eta_{w_1}} \frac{\boldsymbol{\beta}_k^\intercal\boldsymbol{\beta}_k}{||\boldsymbol{\beta}_k||_F^2}, \tag{A.217}$$

$$\mathbf{W}_{1k}^\intercal\mathbf{W}_{1k} = \frac{\sqrt{\lambda_k^2 + 4\eta_{w_2}\eta_{w_1}||\boldsymbol{\beta}_k||_F^2} - \lambda_k}{2\eta_{w_2}} \frac{\boldsymbol{\beta}_k\boldsymbol{\beta}_k^\intercal}{||\boldsymbol{\beta}_k||_F^2}. \tag{A.218}$$

By substituting these expressions into the decompositions $\mathbf{W}_2\mathbf{W}_2^\intercal = \sum_{k=1}^{N_h}\mathbf{W}_{2k}\mathbf{W}_{2k}^\intercal$ and $\mathbf{W}_1^\intercal\mathbf{W}_1 = \sum_{k=1}^{N_h}\mathbf{W}_{1k}^\intercal\mathbf{W}_{1k}$, we derive the representation for $\mathbf{M}$: $\mathbf{M} = \sum_{k=1}^{N_h}\mathbf{M}_k$ where

$$\mathbf{M}_k = \left(\frac{\sqrt{\lambda_k^2 + 4\eta_{w_2}\eta_{w_1}||\boldsymbol{\beta}_k||_F^2} + \lambda_k}{2}\right)\frac{\boldsymbol{\beta}_k^\intercal\boldsymbol{\beta}_k}{||\boldsymbol{\beta}_k||_F^2} \oplus \left(\frac{\sqrt{\lambda_k^2 + 4\eta_{w_2}\eta_{w_1}||\boldsymbol{\beta}_k||_F^2} - \lambda_k}{2}\right)\frac{\boldsymbol{\beta}_k\boldsymbol{\beta}_k^\intercal}{||\boldsymbol{\beta}_k||_F^2}. \tag{A.219}$$

□

Understanding M when there is a single-neuron $N_h = 1$

When there is a single-hidden neuron $N_h = \min(N_i, N_o) = 1$, the expression for $\mathbf{M}$ presented in Theorem 3.4.5 simplifies allowing us to precisely understand the influence of λ on the learning regime. When $N_h = N_o = 1$, then $\frac{\boldsymbol{\beta}^\intercal\boldsymbol{\beta}}{||\boldsymbol{\beta}||_F^2} = 1$. Therefore,

[1] As long as $N_o > 1$, then the surface of this $N_i + N_o$ hyperboloid is always connected, however its topology will depend on the relationship between N_i and N_o.

Eq. 3.28 simplifies to

$$\mathbf{M} = \frac{\sqrt{\lambda^2 + \eta_{w_2}\eta_{w_1}4\|\boldsymbol{\beta}\|^2} + \lambda}{2}\mathbf{I}_{N_i} + \frac{\sqrt{\lambda^2 + \eta_{w_2}\eta_{w_1}4\|\boldsymbol{\beta}\|^2} - \lambda}{2}\frac{\boldsymbol{\beta}\boldsymbol{\beta}^\intercal}{\|\boldsymbol{\beta}\|^2}. \quad (A.220)$$

When $N_h = N_i = 1$, then $\frac{\boldsymbol{\beta}\boldsymbol{\beta}^\intercal}{\|\boldsymbol{\beta}\|_F^2} = 1$ and thus Eq. 3.28 simplifies to,

$$\mathbf{M} = \frac{\sqrt{\lambda^2 + \eta_{w_2}\eta_{w_1}4\|\boldsymbol{\beta}\|^2} + \lambda}{2}\frac{\boldsymbol{\beta}^\intercal\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|^2} + \frac{\sqrt{\lambda^2 + \eta_{w_2}\eta_{w_1}4\|\boldsymbol{\beta}\|^2} - \lambda}{2}\mathbf{I}_{N_o}. \quad (A.221)$$

In both settings, $\mathbf{M}$ is the weighted sum of the identity matrix and a rank-one projection matrix. While these equations are strikingly similar there is an interesting distinction that arises in the limits of λ. As $\lambda \to \infty$, then the first expression for $\mathbf{M}$ becomes proportional to $\mathbf{I}_{N_i}$, while the second expression for $\mathbf{M}$ becomes proportional to the rank-1 projection $\frac{\boldsymbol{\beta}^\intercal\boldsymbol{\beta}}{\|\boldsymbol{\beta}\|^2}$. Conversely, as $\lambda \to -\infty$, then the first expression for $\mathbf{M}$ becomes proportional to the rank-1 projection $\frac{\boldsymbol{\beta}\boldsymbol{\beta}^\intercal}{\|\boldsymbol{\beta}\|^2}$, while the second expression for $\mathbf{M}$ becomes proportional to $\mathbf{I}_{N_o}$. When $N_h = N_i = N_o = 1$, then $\mathbf{M} = \sqrt{\lambda^2 + \eta_{w_2}\eta_{w_1}4\|\boldsymbol{\beta}\|^2}$ and thus in both limits of $\lambda \to \pm\infty$, $\mathbf{M}$ becomes a constant independent of $\boldsymbol{\beta}$. In all settings, when $\lambda = 0$, $\mathbf{M}$ depends on $\boldsymbol{\beta}$. In other words, the influence of λ on whether the dynamics are lazy, rich, or delayed rich, crucially depends on the relative sizes of dimensions N_o, N_h, and N_i.

Interpreting M in different limits and architectures

We now seek to more generally understand the influence of the conserved quantities λ_i and the relative sizes of dimensions N_i, N_h and N_o on the learning regime. For a matrix $\mathbf{W}_2^\intercal \in \mathbb{R}^{N_o \times N_i}$, let $\text{Row}(\mathbf{W}_2^\intercal) \subseteq \mathbb{R}^{N_o}$ and $\text{Col}(\mathbf{W}_2^\intercal) \subseteq \mathbb{R}^{N_i}$ denote the row and column space of $\mathbf{W}_2^\intercal$ respectively.

Theorem A.3.8. *The dynamics are in the lazy regime, for all $t \geq 0$, if $\lambda_k \to \infty$ for all $k \in [N_h]$ and there exists a least squares solution $\boldsymbol{\beta}_* \in \mathbb{R}^{N_i \times N_o}$ such that*

$$\text{Row}(\boldsymbol{\beta}_*) \subseteq \text{Span}\left(\bigcup_{k=1}^{N_h} \text{Row}\left(\boldsymbol{\beta}_k(0)\right)\right), \quad (A.222)$$

or $\lambda_k \to -\infty$ *for all* $k \in [h]$ *and there exists a solution such that*

$$\mathrm{Col}(\boldsymbol{\beta}_*) \subseteq \mathrm{Span}\left(\bigcup_{k=1}^{h} \mathrm{Col}\left(\boldsymbol{\beta}_k(0)\right)\right). \tag{A.223}$$

Proof. As $\lambda_k \to \infty$, $\mathbf{M}_k \to |\lambda_k| \frac{\boldsymbol{\beta}_k^\intercal \boldsymbol{\beta}_k}{\|\boldsymbol{\beta}_k\|_F^2} \otimes \mathbf{I}_d$, implying $\dot{\boldsymbol{\beta}}_k = -|\lambda_k| \frac{\partial \mathcal{L}}{\partial \boldsymbol{\beta}} \left(\frac{\boldsymbol{\beta}_k^\intercal \boldsymbol{\beta}_k}{\|\boldsymbol{\beta}_k\|_F^2}\right)$. Notice that $\left(\frac{\boldsymbol{\beta}_k^\intercal \boldsymbol{\beta}_k}{\|\boldsymbol{\beta}_k\|_F^2}\right)$ is the unique orthogonal projection matrix onto the one-dimensional row space of $\boldsymbol{\beta}_k$. Thus, the dynamics of each $\boldsymbol{\beta}_k$ follow a projected gradient descent in their row space. As a result, $\mathbf{M}_k$ will not change and thus the NTK will be static. By assumption there exists a least squares solution $\boldsymbol{\beta}_*$ such that the rows of $\boldsymbol{\beta}_*$ are in the span of the rows of $\boldsymbol{\beta}_k$. Thus, a solution will be reached as $t \to \infty$, while the $\mathbf{M}_k$ remain static.As $\lambda_k \to -\infty$ for all $k \in [h]$, $\mathbf{M}_k \to \mathbf{I}_c \otimes |\lambda_k| \frac{\boldsymbol{\beta}_k \boldsymbol{\beta}_k^\intercal}{\|\boldsymbol{\beta}_k\|_F^2}$, and an analogous argument can be made. □

Note that the assumptions in Theorem A.3.8 can be more intuitively expressed in terms of the parameter space $(\mathbf{W}_1, \mathbf{W}_2)$. Except in highly degenerate cases, the assumption $\mathrm{Row}(\boldsymbol{\beta}_*) \subseteq \mathrm{Span}\left(\bigcup_{k=1}^{h} \mathrm{Row}\left(\boldsymbol{\beta}_k(0)\right)\right)$ is equivalent to the existence of a $\boldsymbol{\beta}_*$ whose rows lie in the span of $\{\mathbf{W_2 W_1}_k(0)\}_{k=1}^{N_h}$, or, equivalently, to the existence of a matrix $\mathbf{W}_2$ such that $\boldsymbol{\beta}_* = \mathbf{W}_2(0)\mathbf{W}_1$. Similarly, the condition $\mathrm{Col}(\boldsymbol{\beta}_*) \subseteq \mathrm{Span}\left(\bigcup_{k=1}^{N_h} \mathrm{Col}\left(\boldsymbol{\beta}_k(0)\right)\right)$ is in most cases equivalent to the existence of a matrix $\mathbf{W}_2$ such that $\boldsymbol{\beta}_* = \mathbf{W}_1(0)^\intercal \mathbf{W}_2(0)^\intercal$.

A direct consequence of Theorem A.3.8 is that networks which narrow from input to output ($N_i > N_o$) must enter the lazy regime with probability 1 as all $\lambda_k \to \infty$ whenever $N_h \geq N_o$ and assuming independent initialisations for all $\boldsymbol{\beta}_k$. In this case, the rows of $\{\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_h\}$ span all of $\mathbb{R}^{N_o}$ and thus the condition on the least squares solution is trivially true. By the same logic, networks which expand from input to output ($N_i < N_o$) do so as all $\lambda_k \to -\infty$ whenever $N_h \geq N_i$ and assuming independent initialisations for all $\boldsymbol{\beta}_k$. Additionally, when $N_h \geq \max(N_i, N_o)$ and assuming independent initialisations for all $\boldsymbol{\beta}_k$, then all networks enter the lazy regime as either all $\lambda_k \to \infty$ or all $\lambda_k \to -\infty$.

Another interesting implication of Theorem A.3.8, is that if there does not exist a least squares solution $\boldsymbol{\beta}_*$ with rows in the span of the rows of $\{\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_h\}$, then the network will enter a delayed rich regime as all $\lambda_k \to \infty$, where the magnitude of the λ_k will determine the delay. In this setting, the network is initially lazy, attempting to fit the solution within the row space of the $\boldsymbol{\beta}_k$, but eventually the direction of the rows must change in order to fit the problem, leading to a rich phase. A similar statement involving the columns of $\boldsymbol{\beta}_*$ is true as all $\lambda_k \to -\infty$.

A.3.5.3 Simplifying $\mathbf{M}$ through assumptions on $\boldsymbol{\Lambda}$

We now consider how introducing structures on $\boldsymbol{\Lambda}$ can lead to simpler expressions for $\mathbf{M}$. A natural assumption to consider is the following:

A5 (*Isotropic initialisation*). Let $\mathbf{W}_2^\intercal \in \mathbb{R}^{N_h \times N_o}$ and $\mathbf{W}_1 \in \mathbb{R}^{N_h \times N_i}$ be initialised such that $\boldsymbol{\Lambda} = \eta_{w_1}\mathbf{W}_2(0)^\intercal \mathbf{W}_2(0) - \eta_{w_2}\mathbf{W}_1(0)\mathbf{W}_1(0)^\intercal = \lambda \mathbf{I}_{N_h}$.

In *square networks*, where the dimensions of the input, hidden, and output layers coincide ($N_i = N_h = N_o$), and the weights are initialised as $\mathbf{W}_2^\intercal \sim \mathcal{N}(0, \sigma_a^2/N_o)$ and $\mathbf{W}_1 \sim \mathcal{N}(0, \boldsymbol{\Sigma}_{w_1}^2/N_i)$, this assumption is naturally satisfied with $\lambda = \sigma_a^2 - \boldsymbol{\Sigma}_{w_1}^2$ as the dimension $N_h \to \infty$. However, a limitation of this assumption is that for general λ it requires $N_h \leq \min(N_i, N_o)$. Specifically, when $\lambda > 0$, the isotropic initialisation requires that $\mathbf{W}_2(0)^\intercal \mathbf{W}_2(0) \succ 0$, which implies $N_h \leq N_o$. Similarly, when $\lambda < 0$, the isotropic initialisation requires that $\mathbf{W}_1(0)\mathbf{W}_1(0)^\intercal \succ 0$, which implies $N_h \leq N_i$. Now we prove two important implications of the isotropic initialisation assumption.

Lemma A.3.9. *Let $\boldsymbol{\Lambda} = \lambda \mathbf{I}_{N_h}$. If either $\lambda \geq 0$ or $\lambda < 0$ and $N_h \geq N_i$, we have that*

$$\mathbf{W}_1^\intercal \mathbf{W}_1 = \frac{1}{\eta_{w_2}} \left(-\frac{\lambda}{2}\mathbf{I}_{N_i} + \sqrt{\eta_{w_2}\eta_{w_1}\boldsymbol{\beta}\boldsymbol{\beta}^\intercal + \frac{\lambda^2}{4}\mathbf{I}_{N_i}} \right). \tag{A.224}$$

Proof. The quantity $\eta_{w_1}\mathbf{W}_2^\intercal \mathbf{W}_2 - \eta_{w_2}\mathbf{W}_1\mathbf{W}_1^\intercal = \lambda \mathbf{I}_{N_h}$ is conserved in gradient flow. Multiplying on the left by $\mathbf{W}_1^\intercal$ and on the right by $\mathbf{W}_1$ we have that

$$\eta_{w_2}(\mathbf{W}_1^\intercal \mathbf{W}_1)^2 + \lambda \mathbf{W}_1^\intercal \mathbf{W}_1 = \eta_{w_1}\boldsymbol{\beta}\boldsymbol{\beta}^\intercal. \tag{A.225}$$

Completing the square by adding $\frac{\lambda^2}{4\eta_{w_2}}\mathbf{I}_{N_i}$ to both sides and dividing by η_{w_2} we get the equality,

$$\left(\mathbf{W}_1^\intercal\mathbf{W}_1 + \frac{\lambda}{2\eta_{w_2}}\mathbf{I}_{N_i}\right)^2 = \frac{\lambda^2}{4\eta_{w_2}^2}\mathbf{I}_{N_i} + \frac{\eta_{w_1}}{\eta_{w_2}}\boldsymbol{\beta}\boldsymbol{\beta}^\intercal \tag{A.226}$$

For $\lambda \geq 0$, $\mathbf{W}_1^\intercal\mathbf{W}_1 + \frac{\lambda}{2\eta_{w_2}}\mathbf{I}_{N_i} \succeq 0$. For $\lambda < 0$, then we know from the conserved quantity that $\mathbf{W}_1\mathbf{W}_1^\intercal + \frac{\lambda}{2\eta_{w_2}}\mathbf{I}_{N_h} = \frac{\eta_{w_1}}{\eta_{w_2}}\mathbf{W}_2^\intercal\mathbf{W}_2 - \frac{\lambda}{2\eta_{w_2}}\mathbf{I}_{N_h} \succ 0$, which implies when $N_h \geq N_i$ that $\mathbf{W}_1^\intercal\mathbf{W}_1 + \frac{\lambda}{2\eta_{w_2}}\mathbf{I}_{N_i} \succ 0$. As a result, we can take the principal square root of each side,

$$\mathbf{W}_1^\intercal\mathbf{W}_1 + \frac{\lambda}{2\eta_{w_2}}\mathbf{I}_{N_i} = \sqrt{\frac{\lambda^2}{4\eta_{w_2}^2}\mathbf{I}_{N_i} + \frac{\eta_{w_1}}{\eta_{w_2}}\boldsymbol{\beta}\boldsymbol{\beta}^\intercal}, \tag{A.227}$$

which rearranged gives the final result. □

Lemma A.3.10. *Let $\boldsymbol{\Lambda} = \lambda\mathbf{I}_{N_h}$. If either $\lambda \leq 0$ or $\lambda > 0$ and $N_h \geq N_o$, we have that*

$$\mathbf{W}_2\mathbf{W}_2^\intercal = \frac{1}{\eta_{w_1}}\left(\frac{\lambda}{2}\mathbf{I}_{N_o} + \sqrt{\eta_{w_2}\eta_{w_1}\boldsymbol{\beta}^\intercal\boldsymbol{\beta} + \frac{\lambda^2}{4}\mathbf{I}_{N_o}}\right). \tag{A.228}$$

Proof. The proof is analogous to the proof of Lemma A.3.9. □

From Lemma A.3.9 and Lemma A.3.10 we can prove Theorem 3.4.6, as shown below.

Proof. We start from

$$\mathrm{vec}\left(\dot{\boldsymbol{\beta}}\right) = -\underbrace{\left(\eta_{w_1}\mathbf{W}_2\mathbf{W}_2^\intercal \oplus \eta_{w_2}\mathbf{W}_1^\intercal\mathbf{W}_1\right)}_{M}\mathrm{vec}(\mathbf{X}\mathbf{X}^\intercal\boldsymbol{\beta} - \mathbf{X}\mathbf{Y}^T), \tag{A.229}$$

Plugging in expressions for $\mathbf{W}_1^\intercal\mathbf{W}_1$ from Lemma A.3.9 and $\mathbf{W}_2\mathbf{W}_2^\intercal$ from Lemma A.3.10 we can directly write,

$$\begin{aligned}\mathbf{M} &= \left(\frac{\lambda}{2}\mathbf{I}_{N_o} + \sqrt{\eta_{w_2}\eta_{w_1}\boldsymbol{\beta}^\intercal\boldsymbol{\beta} + \frac{\lambda^2}{4}\mathbf{I}_{N_o}}\right) \oplus \left(-\frac{\lambda}{2}\mathbf{I}_{N_i} + \sqrt{\eta_{w_2}\eta_{w_1}\boldsymbol{\beta}\boldsymbol{\beta}^\intercal + \frac{\lambda^2}{4}\mathbf{I}_{N_i}}\right)\\ &= \left(\sqrt{\eta_{w_2}\eta_{w_1}\boldsymbol{\beta}^\intercal\boldsymbol{\beta} + \frac{\lambda^2}{4}\mathbf{I}_{N_o}} \otimes \mathbf{I}_{N_i}\right) + \left(\mathbf{I}_{N_o} \otimes \sqrt{\eta_{w_2}\eta_{w_1}\boldsymbol{\beta}\boldsymbol{\beta}^\intercal + \frac{\lambda^2}{4}\mathbf{I}_{N_i}}\right)\end{aligned} \tag{A.230}$$

□

From this expression for $\mathbf{M}(\boldsymbol{\beta})$ we can easily consider how it simplifies in limiting settings of λ:

$$M \rightarrow \begin{cases} \lambda \mathbf{I}_{N_i N_o} & \lambda \rightarrow -\infty \\ \sqrt{\eta_{w_2}\eta_{w_1}\boldsymbol{\beta}^\intercal\boldsymbol{\beta}} \otimes \mathbf{I}_{N_i} + \mathbf{I}_{N_o} \otimes \sqrt{\eta_{w_2}\eta_{w_1}\boldsymbol{\beta}\boldsymbol{\beta}^\intercal} & \lambda = 0 \\ \lambda \mathbf{I}_{N_i N_o} & \lambda \rightarrow \infty. \end{cases} \tag{A.231}$$

As $\lambda \rightarrow \pm\infty$, $\mathbf{M} \rightarrow \lambda \mathbf{I}_{N_i N_o}$, and the dynamics are lazy. In this limit, the dynamics of $\boldsymbol{\beta}$ converge to the trajectory of linear regression trained by gradient flow and along this trajectory the NTK matrix remains constant. When $\lambda = 0$, $\mathbf{M} = \sqrt{\eta_{w_2}\eta_{w_1}\boldsymbol{\beta}^\intercal\boldsymbol{\beta}} \otimes \mathbf{I}_{N_i} + \mathbf{I}_{N_o} \otimes \sqrt{\eta_{w_2}\eta_{w_1}\boldsymbol{\beta}\boldsymbol{\beta}^\intercal}$, and the dynamics are rich. Here the NTK changes in both magnitude and direction through training. In the next section we will attempt to better understand these dynamics for intermediate values of λ through the lens of a mirror flow.

A.3.5.4 Deriving a mirror flow for the singular values of $\boldsymbol{\beta}$

For a matrix $\boldsymbol{\beta}$, the dynamics of one of its singular values are given by $\dot{\boldsymbol{\sigma}} = \mathbf{v}^\intercal\dot{\boldsymbol{\beta}}\mathbf{u}$, where $\mathbf{u}$ and $\mathbf{v}$ are the corresponding left and right singular vectors as previously defined. This equality can be derived from chain rule and the fact that $||\mathbf{u}|| = ||\mathbf{v}|| = 1$:

$$\dot{\boldsymbol{\sigma}} = \dot{\mathbf{v}}^\intercal\boldsymbol{\beta}\mathbf{u} + \mathbf{v}^\intercal\dot{\boldsymbol{\beta}}\mathbf{u} + \mathbf{v}^\intercal\boldsymbol{\beta}\dot{\mathbf{u}} = \dot{\mathbf{v}}^\intercal\mathbf{u}\boldsymbol{\sigma} + \mathbf{v}^\intercal\dot{\boldsymbol{\beta}}\mathbf{u} + \boldsymbol{\sigma}\mathbf{v}^\intercal\dot{\mathbf{u}} = \mathbf{v}^\intercal\dot{\boldsymbol{\beta}}\mathbf{u}. \tag{A.232}$$

In the last equality we used that fact that for any vector $\mathbf{z}$ with a fixed norm, $\dot{||\mathbf{z}||^2} = 2\dot{\mathbf{z}}^\intercal\mathbf{z} = 0$. Letting $\mathrm{diag}: \mathbb{R}^{N_i \times N_o} \rightarrow \mathbb{R}^{\min(N_i,N_o)}$ be the operator that, given a rectangular matrix, returns a vector of the elements on the main diagonal, we can then write,

$$\dot{\mathbf{S}} = \mathrm{diag}(\mathbf{V}^\intercal\dot{\boldsymbol{\beta}}\mathbf{U}) \tag{A.233}$$

where $\mathbf{S} \in \mathbb{R}^{\min(N_i,N_o)}$ is the vector of singular values of $\boldsymbol{\beta}$. In the following lemma, we use the shared singular vector structure between $\boldsymbol{\beta}$ and $\mathbf{W}_2$ and $\mathbf{W}_1$ to rewrite these dynamics as

$$\dot{\mathbf{S}} = -\mathbf{M}\nabla_{\mathbf{S}}\mathcal{L} \tag{A.234}$$

where $\mathbf{M}$ is a diagonal matrix and $\nabla_{\mathbf{S}}\mathcal{L}$ is the gradient of the loss with respect to the singular values of $\boldsymbol{\beta}$. Without loss of generality we consider $\eta_{w_2} = \eta_{w_1} = 1$.

Lemma A.3.11. *Let $\boldsymbol{\Lambda} = \lambda \mathbf{I}_{N_h}$. We then have that $\dot{\mathbf{S}} = -\mathbf{M}\nabla_{\mathbf{S}}\mathcal{L}$, where $\mathbf{M} \in \mathbb{R}^{\min(N_i,N_o)\times\min(N_o,N_i)}$ is a diagonal matrix with*

$$\mathbf{M}_{ii} = \begin{cases} \sqrt{\lambda^2 + 4\mathbf{S}_i^2} & i \le \min(N_i, N_h, N_o) \\ 0 & \text{otherwise} \end{cases} \tag{A.235}$$

Proof. First note that

$$\begin{aligned} \dot{\mathbf{S}} &= \mathrm{diag}(\mathbf{V}^\intercal \dot{\boldsymbol{\beta}} \mathbf{U}) \\ &= -\mathrm{diag}\left(\mathbf{V}^\intercal \left[\mathbf{X}(\mathbf{X}^\intercal \boldsymbol{\beta} - \mathbf{Y}^\intercal)\mathbf{W}_2\mathbf{W}_2^\intercal + \mathbf{W}_1^\intercal \mathbf{W}_1 \mathbf{X}(\mathbf{X}^\intercal\boldsymbol{\beta} - \mathbf{Y}^\intercal)\right]\mathbf{U}\right) \\ &= -\mathrm{diag}\left(\mathbf{V}^\intercal \mathbf{X}(\mathbf{X}^\intercal\boldsymbol{\beta} - \mathbf{Y}^\intercal)\mathbf{U}\boldsymbol{\Sigma}_{w_2}^2 + \boldsymbol{\Sigma}_{w_1}^2 \mathbf{V}^\intercal\mathbf{X}(\mathbf{X}^\intercal\boldsymbol{\beta} - \mathbf{Y}^\intercal)\mathbf{U}\right) \end{aligned} \tag{A.236}$$

where we let $\mathbf{W}_1^\intercal\mathbf{W}_1 = \mathbf{V}\boldsymbol{\Sigma}_{w_1}^2\mathbf{V}^\intercal$ and $\mathbf{W}_2\mathbf{W}_2^\intercal = \mathbf{U}\boldsymbol{\Sigma}_{w_2}^2\mathbf{U}^\intercal$, using the fact that, under $\boldsymbol{\Lambda} = \lambda\mathbf{I}_{N_h}$, the eigenvectors of $\mathbf{W}_2\mathbf{W}_2^\intercal$ are the right singular vectors of $\boldsymbol{\beta}$ and the eigenvectors of $\mathbf{W}_1^\intercal\mathbf{W}_1$ are the left singular vectors of $\boldsymbol{\beta}$. This expression rewrites as

$$\dot{\mathbf{S}} = -\mathbf{M}\mathrm{diag}\left(\mathbf{V}^\intercal\mathbf{X}(\mathbf{X}^\intercal\boldsymbol{\beta} - \mathbf{Y}^\intercal)\mathbf{U}\right) \tag{A.237}$$

where $\mathbf{M} \in \mathbb{R}^{\min(N_i,N_o)\times\min(N_i,N_o)}$ is a diagonal matrix with $\mathbf{M}_{ii} = (\boldsymbol{\Sigma}_{w_2}^2)_{ii} + (\boldsymbol{\Sigma}_{w_1}^2)_{ii}$. For $i \le \min(N_i, N_h, N_o)$, one can show that $\mathbf{M}_{ii} = \sqrt{\lambda^2 + 4\mathbf{S}_i^2}$. This is because for $i \le \min(N_i, N_h, N_o)$, $(\boldsymbol{\Sigma}_{w_2}^2)_{ii} = (\boldsymbol{\Sigma}_{w_1}^2)_{ii} + \lambda$ from the conservation law and $(\boldsymbol{\Sigma}_{w_1}^2)_{ii}(\boldsymbol{\Sigma}_{w_2}^2)_{ii} = \mathbf{S}_i^2$ from the definition of $\mathbf{S}$. Together this implies $(\boldsymbol{\Sigma}_{w_1}^2)_{ii}\left(\lambda + (\boldsymbol{\Sigma}_{w_1}^2)_{ii}\right) = \mathbf{S}_i^2$, which is a quadratic equation in $(\boldsymbol{\Sigma}_{w_1}^2)_{ii}$. If $N_h < \min(N_i, N_o)$ then $\mathbf{M}_{ii} = 0$ for $i > \min(N_i, N_o)$ accounting for rank deficiency of both $\mathbf{W}_2^T$ and $\mathbf{W}_1$ in this case. Additionally, in our setting of MSE loss, it is straightforward to show that

$$\frac{\partial\mathcal{L}}{\partial\mathbf{S}_i} = \left(\mathbf{V}^\intercal\mathbf{X}(\mathbf{X}^\intercal\boldsymbol{\beta} - \mathbf{Y}^\intercal)\mathbf{U}\right)_{ii} \tag{A.238}$$

We then have that $\nabla_{\mathbf{S}}\mathcal{L} = \operatorname{diag}(\mathbf{V}^\intercal \mathbf{X}(\mathbf{X}^\intercal \boldsymbol{\beta} - \mathbf{Y}^\intercal)\mathbf{U})$, which, combined with our expression for $\mathbf{M}$, completes the proof. □

Leveraging Lemma A.3.11, we can show that the singular values of $\boldsymbol{\beta}$ evolve under a mirror flow in the following theorem.

Theorem A.3.12. *Let $\boldsymbol{\Lambda} = \lambda \mathbf{I}_{N_h}$ and assume $N_h \geq \min(N_i, N_o)$ and $\mathbf{S} \neq 0$. We then have that the dynamics of $\mathbf{S}$, the singular values of $\boldsymbol{\beta}$, are given by the mirror flow*

$$\dot{\mathbf{S}} = -\left(\nabla^2 \Phi_\lambda(\mathbf{S})\right)^{-1} \nabla_{\mathbf{S}}\mathcal{L}, \tag{A.239}$$

where $\Phi_\lambda(\mathbf{S}) = \sum_{i=1}^{\min(N_i,N_o)} q_\lambda(\mathbf{S}_i)$ and q_λ is the hyperbolic entropy potential

$$q_\lambda(x) = \frac{1}{4}\left(2x \sinh^{-1}\left(\frac{2x}{|\lambda|}\right) - \sqrt{4x^2 + \lambda^2} + |\lambda|\right). \tag{A.240}$$

Proof. When $\boldsymbol{\Lambda} = \lambda \mathbf{I}_{N_h}$, then by Lemma A.3.11 the dynamics of the singular values of $\boldsymbol{\beta}$ can be expressed as $\dot{\mathbf{S}} = -\mathbf{M}\nabla_{\mathbf{S}}\mathcal{L}$. Furthermore, when $N_h \geq \min(N_i, N_o)$ and $\mathbf{S} \neq 0$, we have that $\mathbf{M} = \sqrt{\lambda^2 + 4\mathbf{S}^2}\mathbf{I}_{\min(N_i,N_o)}$, where $\mathbf{S}^2$ is element-wise, which is always invertible. Observe, this expression for $\mathbf{M}$ is the inverse Hessian of the potential $\Phi_\lambda(\mathbf{S}) = \sum_i q_\lambda(\mathbf{S}_i)$ for q_λ specified in the theorem statement. Thus, the dynamics for the singular values are the mirror flow $\dot{\mathbf{S}} = -\left(\nabla^2 \Phi_\lambda(\mathbf{S})\right)^{-1} \nabla_{\mathbf{S}}\mathcal{L}$.

□

Theorem A.3.12 implies that the dynamics for the singular values of $\boldsymbol{\beta}$ can be described as a mirror flow with a λ-dependent potential. This potential was first identified as the inductive bias for diagonal linear networks by Woodworth et al. [294]. Termed *hyperbolic entropy*, this potential smoothly interpolates between an ℓ^1 and ℓ^2 penalty on the singular values for the rich ($\lambda \to 0$) and lazy ($\lambda \to \pm\infty$) regimes respectively. Unfortunately, in our setting we cannot adapt our mirror flow interpretation into a statement on the inductive bias at interpolation because the singular vectors evolve through training. If we introduce additional assumptions — specifically, whitened input data ($\mathbf{X}\mathbf{X}^\intercal = \mathbf{I}_{N_i}$) and a task-aligned initialisation such that the singular vectors of $\boldsymbol{\beta}_0$ are aligned with those of $\boldsymbol{\beta}_*$ — we can ensure that the singular vectors remain constant and thus derive an inductive bias on the singular values. However, in this setting the dynamics decouple completely, implying there is no difference between applying an ℓ^1 or ℓ^2 penalty on the singular values. Consequently, even though the dynamics will depend on λ, the final interpolating solution will be independent of λ, making a statement on the inductive bias insignificant.

A.3.5.5 Delayed Rich

Theorem A.3.13. *Under the conditions of Theorem A.2.6, when $\lambda_\perp > 0$, the network enters a regime referred to as the delayed-rich phase. In this phase, the learning rate is determined by two competing exponential factors:*

$$e^{\lambda_\perp \frac{t}{\tau}} e^{-\sqrt{\tilde{\mathbf{S}}^2+\frac{\lambda^2}{4}\mathbf{I}}\frac{t}{\tau}} \quad \textit{and} \quad e^{-\sqrt{\tilde{\mathbf{S}}^2+\frac{\lambda^2}{4}\mathbf{I}}\frac{t}{\tau}}. \tag{A.241}$$

As λ increases, various parts of the network display different learning dynamics: some components adjust rapidly, converging exponentially with λ, while others adapt more slowly, with their convergence rate inversely proportional to λ, leading to a slow adaptation.

Proof. The solution to Theorem A.2.6 is governed by two time-dependent terms:

$$e^{-\sqrt{\tilde{\mathbf{S}}^2+\frac{\lambda^2\mathbf{I}}{4}}\frac{t}{\tau}} \quad \text{and} \quad e^{\lambda_\perp \frac{t}{\tau}} e^{-\sqrt{\tilde{\mathbf{S}}^2+\frac{\lambda^2}{4}\mathbf{I}}\frac{t}{\tau}}. \tag{A.242}$$

The first term decays exponentially, tending toward zero over time.

$$\lim_{t\to\infty} e^{-\sqrt{\tilde{\mathbf{S}}^2+\frac{\lambda^2\mathbf{I}}{4}}\frac{t}{\tau}} = \mathbf{0}. \tag{A.243}$$

In the limit of λ gets large the rate of learning is given by

$$\lim_{\lambda\to\infty} \frac{\sqrt{\lambda^2\mathbf{I}+4\tilde{\mathbf{S}}^2}}{2} = \frac{\lambda\mathbf{I}}{2}, \tag{A.244}$$

The second term also decays with time

$$\lim_{t\to\infty} e^{\lambda_\perp \frac{t}{\tau}} e^{-\sqrt{\tilde{\mathbf{S}}^2+\frac{\lambda^2}{4}\mathbf{I}}\frac{t}{\tau}} = \mathbf{0}, \tag{A.245}$$

but in the limit as lambda gets large the rate of learning is given by

$$\lim_{\lambda\to\infty} \frac{\sqrt{\lambda^2\mathbf{I}+4\tilde{\mathbf{S}}^2}-\lambda\mathbf{I}}{2} = \frac{\tilde{\mathbf{S}}^2}{\lambda}. \tag{A.246}$$

Thus, as λ increases, the convergence rate slows for certain parts of the network, while other components continue to learn more quickly. This explains the delay observed in the delayed-rich regime.

□

A.3.5.6 Analysing the impact of recurrence on feature learning

Here we derive the finite-width neural tangent kernel (NTK) [130] for a LRNN where the loss is computed over the final output of the network, following a similar approach to Braun et al. [39], Dominé et al. [69] in deep linear networks.

Recall the network function of the LRNN at training step t_θ is

$$\hat{\boldsymbol{Y}}_{T,t_\theta}(\boldsymbol{X}_{1:T}) = \mathbf{W}_2 \sum_{i=1}^{T} \mathbf{W}_h^{T-i}\mathbf{W}_1\boldsymbol{X}_i. \tag{A.247}$$

After taking a training step with learning rate η, the network function becomes

$$\hat{\boldsymbol{Y}}_{T,t_\theta+1}(\boldsymbol{X}_{1:T}) = (\mathbf{W}_2-\eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_2})\sum_{i=1}^{T}(\mathbf{W}_h-\eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_h})^{T-i}(\mathbf{W}_1-\eta\frac{\partial\mathcal{L}}{\partial W_1})\boldsymbol{X}_i \tag{A.248}$$

Using the binomial expansion $(a-b)^n = \sum_{k=0}^{n}(-1)^k\binom{n}{k}a^{n-k}b^k$

$$\begin{aligned}(\mathbf{W}_h-\eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_h})^{T-i} &= \sum_{k=0}^{T-i}(-1)^k\binom{T-i}{k}\mathbf{W}_h^{T-i-k}\left(\eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_h}\right)^k\\ &= \mathbf{W}_h^{T-i}+\sum_{k=1}^{T-i}(-1)^k\binom{T-i}{k}\mathbf{W}_h^{T-i-k}\left(\eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_h}\right)^k.\end{aligned} \tag{A.249}$$

Substituting back,

$$
\begin{aligned}
&\hat{\boldsymbol{Y}}_{T,t_\theta+1}(\boldsymbol{X}_{1:T}) = \\
&(\mathbf{W}_2 - \eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_2})\sum_{i=1}^{T}\left(\mathbf{W}_h^{T-i} + \sum_{k=1}^{T-i}(-1)^k\binom{T-i}{k}\mathbf{W}_h^{T-i-k}(\eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_h})^k\right) \\
&\quad\times(\mathbf{W}_1 - \eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_1})\boldsymbol{X}_i \\
&= \sum_{i=1}^{T}\Bigg[\mathbf{W}_2\mathbf{W}_h^{T-i}\mathbf{W}_1\boldsymbol{X}_i + \mathbf{W}_2\sum_{k=1}^{T-i}(-1)^k\binom{T-i}{k}\mathbf{W}_h^{T-i-k}(\eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_h})^k\mathbf{W}_1\boldsymbol{X}_i \\
&\quad-\mathbf{W}_2\eta\mathbf{W}_h^{T-i}\frac{\partial\mathcal{L}}{\partial\mathbf{W}_1}\boldsymbol{X}_i - \mathbf{W}_2\eta\sum_{k=1}^{T-i}(-1)^k\binom{T-i}{k}\mathbf{W}_h^{T-i-k}(\eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_h})^k\frac{\partial\mathcal{L}}{\partial\mathbf{W}_1}\boldsymbol{X}_i \\
&\quad-\eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_2}\mathbf{W}_h^{T-i}\mathbf{W}_1\boldsymbol{X}_i - \eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_2}\sum_{k=1}^{T-i}(-1)^k\binom{T-i}{k}\mathbf{W}_h^{T-i-k}(\eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_h})^k\mathbf{W}_1\boldsymbol{X}_i \\
&\quad+\eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_2}\eta\mathbf{W}_h^{T-i}\frac{\partial\mathcal{L}}{\partial\mathbf{W}_1}\boldsymbol{X}_i + \eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_2}\eta\sum_{k=1}^{T-i}(-1)^k\binom{T-i}{k}\mathbf{W}_h^{T-i-k}(\eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_h})^k\frac{\partial\mathcal{L}}{\partial\mathbf{W}_1}\boldsymbol{X}_i\Bigg].
\end{aligned}
\tag{A.250}
$$

The gradient flow equation describing the dynamics of the network function is then:

$$
\begin{aligned}
&\frac{\hat{\boldsymbol{Y}}_{T,t_\theta+1} - \hat{\boldsymbol{Y}}_{T,t_\theta}}{\eta} = \\
&\sum_{i=1}^{T}\Bigg[\mathbf{W}_2\sum_{k=1}^{T-i}(-1)^k\binom{T-i}{k}\mathbf{W}_h^{T-i-k}(\eta)^{k-1}(\frac{\partial\mathcal{L}}{\partial\mathbf{W}_h})^k\mathbf{W}_1\boldsymbol{X}_i \\
&\quad-\mathbf{W}_2\mathbf{W}_h^{T-i}\frac{\partial\mathcal{L}}{\partial\mathbf{W}_1}\boldsymbol{X}_i - \mathbf{W}_2\sum_{k=1}^{T-i}(-1)^k\binom{T-i}{k}\mathbf{W}_h^{T-i-k}(\eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_h})^k\frac{\partial\mathcal{L}}{\partial\mathbf{W}_1}\boldsymbol{X}_i \\
&\quad-\frac{\partial\mathcal{L}}{\partial\mathbf{W}_2}\sum_{i=1}^{T}\mathbf{W}_h^{T-i}\mathbf{W}_1\boldsymbol{X}_i - \frac{\partial\mathcal{L}}{\partial\mathbf{W}_2}\sum_{k=1}^{T-i}(-1)^k\binom{T-i}{k}\mathbf{W}_h^{T-i-k}(\eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_h})^k\mathbf{W}_1\boldsymbol{X}_i \\
&\quad+\frac{\partial\mathcal{L}}{\partial\mathbf{W}_2}\eta\mathbf{W}_h^{T-i}\frac{\partial\mathcal{L}}{\partial\mathbf{W}_1}\boldsymbol{X}_i + \frac{\partial\mathcal{L}}{\partial\mathbf{W}_2}\eta\sum_{k=1}^{T-i}(-1)^k\binom{T-i}{k}\mathbf{W}_h^{T-i-k}(\eta\frac{\partial\mathcal{L}}{\partial\mathbf{W}_h})^k\frac{\partial\mathcal{L}}{\partial\mathbf{W}_1}\boldsymbol{X}_i\Bigg].
\end{aligned}
\tag{A.251}
$$

As the learning rate $\eta \to 0$ (the gradient flow regime),

$$\tau\frac{\hat{\boldsymbol{Y}}_T}{dt_\theta}=\sum_{i=1}^{T}(-\mathbf{W}_2\mathbf{W}_h^{T-i}\frac{\partial\mathcal{L}}{\partial\mathbf{W}_1}-(T-i)\mathbf{W}_2\mathbf{W}_h^{T-i-1}\frac{\partial\mathcal{L}}{\partial\mathbf{W}_h}\mathbf{W}_1-\frac{\partial\mathcal{L}}{\partial\mathbf{W}_2}\mathbf{W}_h^{T-i}\mathbf{W}_1)\boldsymbol{X}_i. \tag{A.252}$$

Substituting the partial derivatives of the loss,

$$\begin{aligned}\tau\frac{\hat{\boldsymbol{Y}}_T}{dt_\theta}=\sum_{i=1}^{T}-\mathbf{W}_2\mathbf{W}_h^{T-i}\left(\sum_{j=1}^{T}\mathbf{W}_h^{(T-j)\top}\mathbf{W}_2^\top(\boldsymbol{Y}_T-\mathbf{W}_2\sum_{k=1}^{T}\mathbf{W}_h^{T-k}\mathbf{W}_1\boldsymbol{X}_k)\boldsymbol{X}_j^\top\right)\boldsymbol{X}_i\\
-(T-i)\mathbf{W}_2\mathbf{W}_h^{T-i-1}\left(\sum_{j=1}^{T-1}\sum_{r=0}^{T-j-1}\mathbf{W}_h^{(r)\top}\mathbf{W}_2^\top(\boldsymbol{Y}_T-\mathbf{W}_2\sum_{k=1}^{T}\mathbf{W}_h^{T-k}\mathbf{W}_1\boldsymbol{X}_k)\right.\\
\left.\times\boldsymbol{X}_j^\top\mathbf{W}_1^\top\mathbf{W}_h^{(T-j-1-r)\top}\mathbf{W}_1\boldsymbol{X}_i\right)\\
-\left((\boldsymbol{Y}_T-\mathbf{W}_2\sum_{k=1}^{T}\mathbf{W}_h^{T-k}\mathbf{W}_1\boldsymbol{X}_k)\left(\sum_{j=1}^{T}\boldsymbol{X}_j^\top\mathbf{W}_1^\top\mathbf{W}_h^{(T-j)\top}\right)\right)\\
\times\mathbf{W}_h^{T-i}\mathbf{W}_1\boldsymbol{X}_i.\end{aligned} \tag{A.253}$$

Finally, we use the identity $\mathrm{vec}(\mathbf{AXB})=(\mathbf{B}^\top\otimes\mathbf{A})\mathrm{vec}(\mathbf{X})$ to derive the NTK $(\nabla_\theta\mathrm{vec}(\hat{\boldsymbol{Y}}_T)\nabla_\theta\mathrm{vec}(\hat{\boldsymbol{Y}}_T))$ on the left-side of the vectorising function:

$$\begin{aligned}\tau\frac{d\mathrm{vec}(\hat{\boldsymbol{Y}}_T)}{dt_\theta}=&\sum_{i=1}^{T}\sum_{j=1}^{T-1}(-\boldsymbol{X}_i^\top\boldsymbol{X}_j\otimes\mathbf{W}_2\mathbf{W}_h^{T-i}\mathbf{W}_h^{(T-j)\top}\mathbf{W}_2^\top)\\
&-\sum_{r=0}^{T-i-1}(T-i)\boldsymbol{X}_i^\top\mathbf{W}_1^\top\mathbf{W}_h^{(T-k-1-r)}\mathbf{W}_1\boldsymbol{X}_j\\
&\otimes\mathbf{W}_2\mathbf{W}_h^{T-i-1}\mathbf{W}_h^{(r)\top}\mathbf{W}_2^\top\\
&-I_{N_y}\otimes\boldsymbol{X}_i^\top\mathbf{W}_1^\top\mathbf{W}_h^{(T-i)\top}\mathbf{W}_h^{(T-j)}\mathbf{W}_1\boldsymbol{X}_j\\
&\times\mathrm{vec}(\boldsymbol{Y}_T-\mathbf{W}_2\sum_{k=1}^{T}\mathbf{W}_h^{T-k}\mathbf{W}_1\boldsymbol{X}_k).\end{aligned} \tag{A.254}$$

$$\begin{aligned}\nabla_\theta \mathrm{vec}(\hat{\boldsymbol{Y}}_T)\nabla_\theta \mathrm{vec}(\hat{\boldsymbol{Y}}_T) = &\sum_{i=1}^{T}\sum_{j=1}^{T-1}\boldsymbol{X}_i^\top \boldsymbol{X}_j \otimes \mathbf{W}_2\mathbf{W}_h^{T-i}\mathbf{W}_h^{(T-j)\top}\mathbf{W}_2^\top \\ &+ \sum_{r=0}^{T-i-1}(T-i)\boldsymbol{X}_i^\top \mathbf{W}_1^\top \mathbf{W}_h^{(T-k-1-r)}\mathbf{W}_1\boldsymbol{X}_j \\ &\otimes \mathbf{W}_2\mathbf{W}_h^{T-i-1}\mathbf{W}_h^{(r)\top}\mathbf{W}_2^\top \\ &+ I_{N_y}\otimes \boldsymbol{X}_i^\top \mathbf{W}_1^\top \mathbf{W}_h^{(T-i)\top}\mathbf{W}_h^{(T-j)}\mathbf{W}_1\boldsymbol{X}_j. \qquad \text{(A.255)}\end{aligned}$$

We conduct an experiment to investigate how sequence length T (recurrence) and the scale of initialisation weights influence the learning dynamics of the network (Fig. 3.8). All layers are initialised with weights drawn from a normal distribution with a specified variance (initialisation scale). The network is trained on a random input-output mapping sampled from a normal distribution with a variance of 1. We use a learning rate of $\eta = 0.1$ throughout the experiments. In the first experiment, the network is unaligned with the task eigenvectors. In the second experiment, we achieve task alignment by initialising the network with the eigenvectors of the task. During training, we calculate the NTK distance from its initialisation. This kernel distance, $\mathbf{K}(t)$, is defined as:

$$\mathbf{K}(t) = 1 - \frac{\langle \mathbf{K}_0, \mathbf{K}_t\rangle}{\|\mathbf{K}_0\|_F\|\mathbf{K}_t\|_F},$$

following the definition in Fort et al. [91]. These distances are visualised as heatmaps for sequence lengths within the range $[1,7]$ and initialisation scales in $(0.01,1]$. A batch size of $P = 50$ is used for training. As explained in the main text, the recurrence always leads to a rich regime in the unaligned case, regardless of the scale of initialisation. This contrasts with feedforward networks. Under the aligned configuration, vector rotation to align with the task is unnecessary; instead, the scaling of the initialisation becomes the critical factor. When the network's initial variance is close to the target variance, the adjustments required to fit the target are minimised. This explains why smaller initialisation scales result in more pronounced NTK movement. It is important to emphasise that these observations are specific to the final stages of the learning process.

Note that, in this study, we focus on finite-width neural networks, but discuss the connection between our work and from the infinite-width literature. Studies of the learning regimes and dynamics of neural networks have focused on how the variance of initialisation and layer-wise learning rates should scale in the context of infinite-width networks. This scaling is crucial for ensuring consistent behaviour across activations, gradients, and outputs, enabling the network to effectively perform kernel regression without learning specific features [196, 231, 260, 232, 130, 165]. The hyper-parameters for Fig. A.3 are identical to those used in Fig. 3.8, with the sole difference being that the hidden layer size is set to 300 instead of 4. In this context, interestingly, recurrence does not always protect against a lazy regime, corroborating results previously reported in the literature.

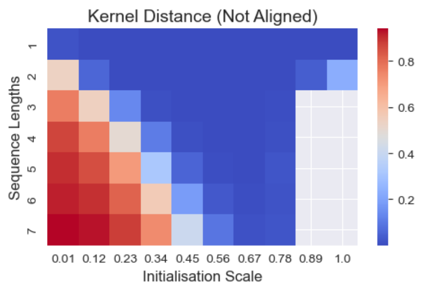


Figure A.3: Recurrence does not drives feature learning in infinite width. Phase plots illustrating the kernel distance of the NTK from initialisation when the hidden layer size is set to 300 compared to size 4.

.

In Fig. A.4, we examine how the trajectory length and the initialisation scale of input-output modes versus recurrent modes affect the learning regime in an aligned setting. As we saw before, increasing the trajectory length tends to favour a richer learning regime. Furthermore, initialisation scale of the recurrent modes more strongly influences the feature learning regime compared to the scale of input-output modes.

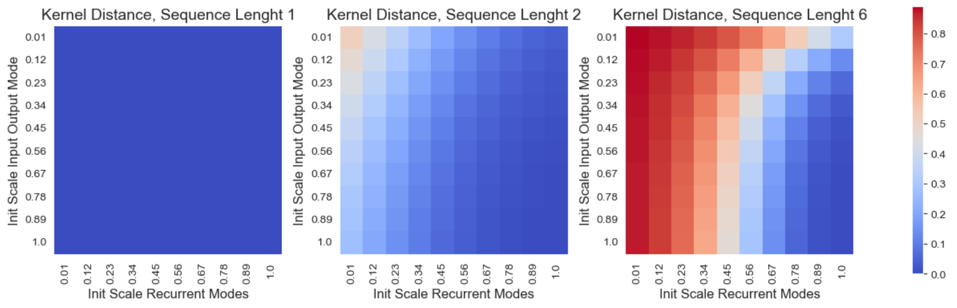


Figure A.4: Initialisation of recurrent modes more strongly impacts feature learning. Phase plots illustrating the kernel distance of the NTK from initialisation for different trajectory lengths as a function of initialisation strength of input-output and recurrent connectivity modes. Initialisation scale of recurrent modes more strongly influences kernel distance than the scale of input-output modes.

.

A.3.6 The impact of scale

A.3.6.1 Scale vs relative scale

We conducted an experiment to explore the relationship between relative weight scale, absolute weight scale, and the network's learning regime in a general setting as shown in Fig. 3.9. The absolute scale of the weights in Fig. 3.9A is defined as the norm of $\mathbf{W}_2\mathbf{W}_1$. Random initial weights with specified relative and absolute scales were generated, and the network was trained on a random input-output task. We compute the logarithmic kernel distance of the NTK from initialisation and the logarithmic loss throughout training. We plot these values in a heat map for λ in $[-9,9]$ and relative scale in $(0,20]$. We repeat this procedure for three architectures: a square network $(N_i = 2, N_h = 2, N_o = 2)$, a funnel network $(N_i = 4, N_h = 2, N_o = 2)$, and an anti-funnel network $(N_i = 2, N_h = 2, N_o = 4)$. These are the same architectures as in Fig. 3.6 in the main text.

Initialisation In all three different types of networks, at the start of training, the model loss depends entirely on an absolute scale, not the relative scale.

Throughout training Across networks, the learning dynamics are intricately influenced by both the absolute and relative scales and fully captured by our theoretical solution. In the square network, the loss increases with absolute scale but

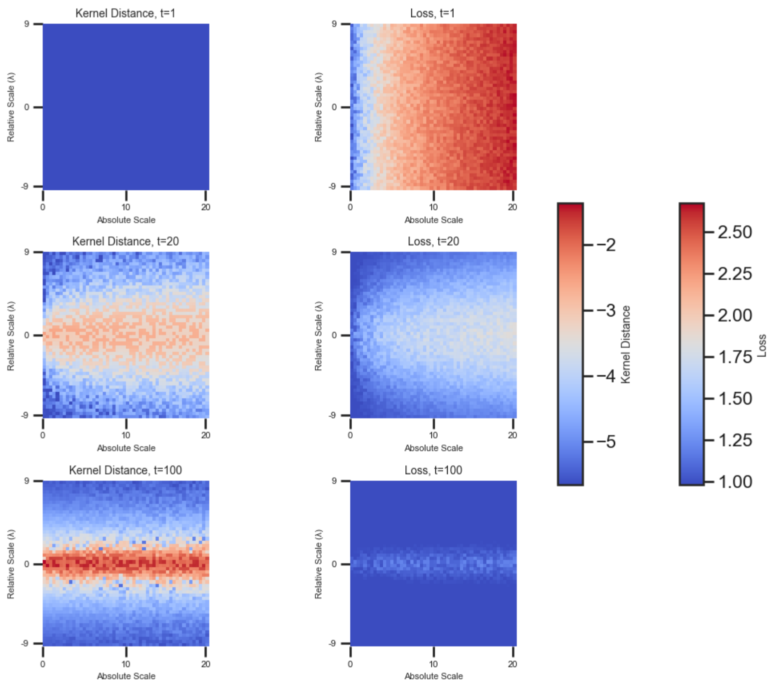


Figure A.5: Square network: Phase plots showing (left) the logarithmic kernel distance of the NTK from initialisation and (right) the corresponding logarithmic loss as functions of the relative scale λ and the absolute scale. (Top to bottom) Different time steps during training $t = 1$, $t = 20$, $t = 100$.

decreases with relative scale, as shown in Fig. A.5. Strikingly, for large imbalanced λ, even at small scales, the network transitions into a lazy regime. The funnel and anti-funnel network architectures demonstrate antisymmetric behaviour as shown in Fig. A.6 and Fig. A.7. Here, we focus on the anti-funnel network for brevity. The evolution of the loss reveals that negative λ initialisations first converge, whereas positive λ initialisations retain larger loss values. Additionally, the kernel distance attains its maximum for positive λ, aligning with the results outlined in Sec. 3.4.

At convergence, the loss across all networks stabilises uniformly, irrespective of initial conditions, confirming consistent convergence. This outcome aligns with the theoretical expectation for linear networks under gradient flow, which predictably converge to the same solution. Furthermore, in square networks, the kernel distance peaks at $\lambda = 0$ (results corresponding to the green curve in Fig. 3.6B.) This observation illustrates that the regime at $\lambda = 0$ is consistently rich, independent of the absolute scale as predicted by our theoretical results in Theorem A.3.4. For funnel and anti-funnel networks, the kernel distance exhibits an antisymmetric pattern. In the anti-funnel network, the kernel distance depends mostly on λ, achieving high values for positive λ and approaching zero for negative λ (matching the results in Fig. 3.6B (pink line)). Conversely, in the funnel network, the kernel distance is high for negative λ and approaches zero for positive λ, corroborating the results in Fig. 3.6B. (blue line). These results emphasise the interplay between relative and absolute scales, highlighting their critical roles in determining the system's behavior. Altogether, the absolute scale and relative scale of the weights play a critical role in describing the phase portrait of the learning regime, as first demonstrated in Kunin et al. [158].

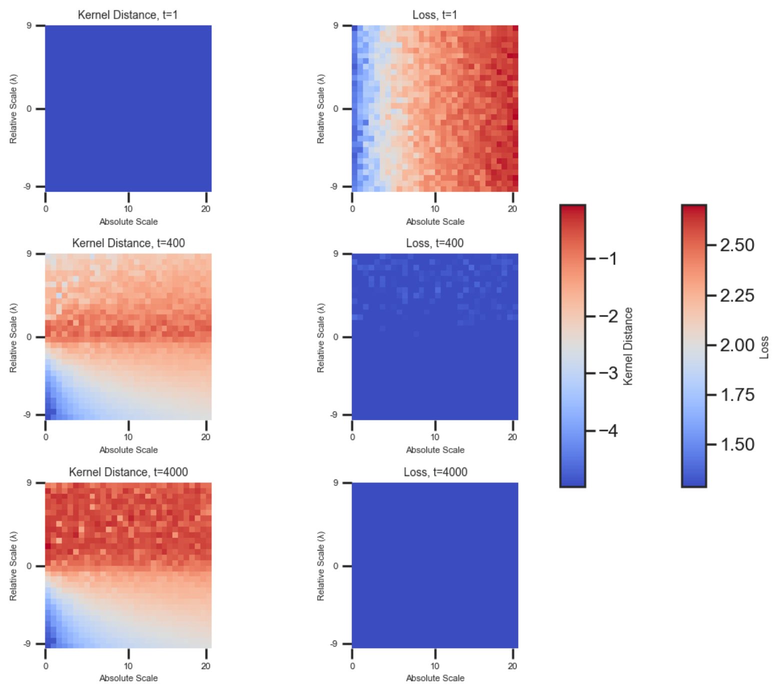


Figure A.6: Anti-funnel network: Phase plots showing (left) the logarithmic kernel distance of the NTK from initialisation and (right) the corresponding logarithmic loss as functions of the relative scale λ and the absolute scale. (Top to bottom) Different time steps during training $t = 1$, $t = 400$, $t = 4000$.

A.3.6.2 Rapid Rich regime

Under the assumptions of Theorem A.3.4, the network function acquires a rich task-specific internal representation at convergence, that is $\mathbf{W}_1^T\mathbf{W}_1 = \tilde{\mathbf{V}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T$ and

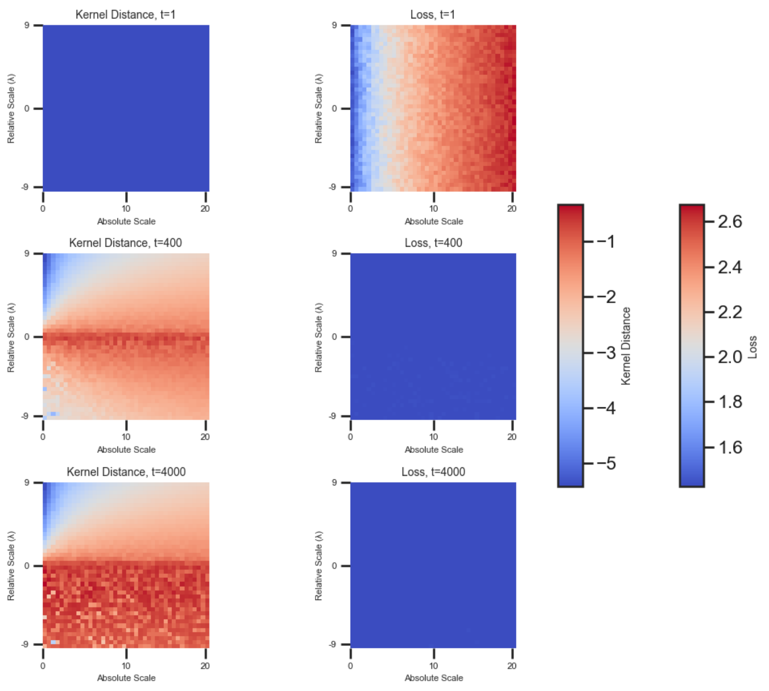


Figure A.7: Funnel network: Phase plots showing (left) the logarithmic kernel distance of the NTK from initialisation and (right) the corresponding logarithmic loss as functions of the relative scale λ and the absolute scale. (Top to bottom) Different time steps during training $t = 1$, $t = 400$, $t = 4000$.

$\mathbf{W}_2\mathbf{W}_2^T = \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{U}}^T$. Therefore, there exist initial states with large zero-balanced weights that lead to rich solutions.

We more quantitatively capture this phenomena in Fig. A.8. We define the error on the internal representation $||\mathbf{W}_1^T\mathbf{W}_1 - \tilde{\mathbf{V}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T||_F^2$ and $||\mathbf{W}_2\mathbf{W}_2^T - \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{U}}^T|_F^2$ for $\mathbf{W}_1$ and $\mathbf{W}_2$ respectively. Effectively, we measure the richness of the representation and in turn it's generalisation ability. In Fig. A.8, the error remains zero for increasing the gain for any network initialised with zero-balanced weights. In other words, the representation at convergence is rich. In contrast, for random initialisation the error increases with increasing gain. As the network is moving away from the small random weight initialisation, the network converges to lazier representation.

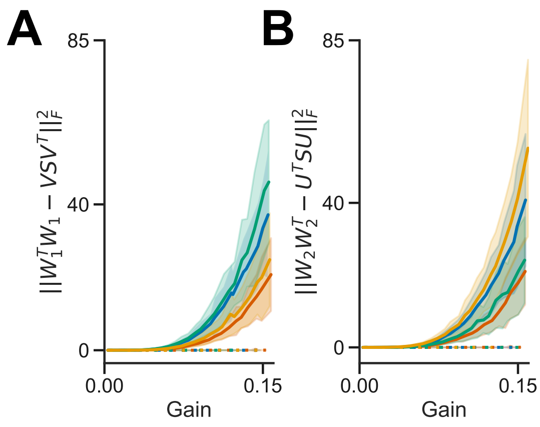


Figure A.8: **A.B** Mean and standard deviation on the error on the internal representation error defined as in Sec. A.3.6.2 for the learning the living kingdom task (Fig. 3.10A), a random 7×7 matrix (blue), a random 5×7 matrix (yellow), a 7×5 matrix (green), a 8×8 matrix (red). All the task ran were ran with learning rate $\eta = 0.001$ enforcing initial zero-balanced weights (Assumption 4) (dotted line) and breaking the assumption of zero-balanced initial weights (line). $N_h = 10$ for all networks.

A.4 Applications

A.4.1 Continual learning

We consider the case of training a two-layer deep linear network on a sequence of tasks $\mathcal{T}_a$, $\mathcal{T}_b$, $\mathcal{T}_c$, ... with corresponding correlation functions $\mathcal{T}_a = \tilde{\mathbf{\Sigma}}_a^{yx}$, $\mathcal{T}_b = \tilde{\mathbf{\Sigma}}_b^{yx}$ .... Then, the full batch loss of the i-th task at any point in training time is

$$\mathcal{L}_i = \frac{1}{2P}||\mathbf{W}_2\mathbf{W}_1\mathbf{X}_i - \mathbf{Y}_i||_F^2 . \tag{A.256}$$

From Theorem A.3.3 it follows that after training the network to convergence on task $\mathcal{T}_j$, the network function is $\mathbf{W}_2\mathbf{W}_1 = \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T = \tilde{\mathbf{\Sigma}}_j^{yx}$. Further, using the assumption of whitened inputs (Assumption 1) and the identities $||\mathbf{A}||_F^2 = Tr(\mathbf{A}\mathbf{A}^T)$ and $Tr(\mathbf{A}) + Tr(\mathbf{B}) = Tr(\mathbf{A}+\mathbf{B})$, the full batch loss of the i-th task is then

$$
\begin{aligned}
\mathcal{L}_i\left(\mathcal{T}_j\right) &= \frac{1}{2P}\|\mathbf{W}_2\mathbf{W}_1\mathbf{X}_i - \mathbf{Y}_i\|_F^2 \\
&= \frac{1}{2P}\operatorname{Tr}\left((\mathbf{W}_2\mathbf{W}_1\mathbf{X}_i - \mathbf{Y}_i\,|)(\mathbf{W}_2\mathbf{W}_1\mathbf{X}_i - \mathbf{Y}_i\,|)^T\right) \\
&= \frac{1}{2P}\operatorname{Tr}\left(\mathbf{W}_2\mathbf{W}_1\mathbf{X}_i\mathbf{X}_i^T(\mathbf{W}_2\mathbf{W}_1)^T\right) - \frac{1}{P}\operatorname{Tr}\left(\mathbf{W}_2\mathbf{W}_1\mathbf{X}_i\mathbf{Y}_i^T\right) + \frac{1}{2P}\operatorname{Tr}\left(\mathbf{Y}_i\mathbf{Y}_i^T\right) \\
&= \frac{1}{2}\operatorname{Tr}\left(\mathbf{W}_2\mathbf{W}_1(\mathbf{W}_2\mathbf{W}_1)^T\right) - \operatorname{Tr}\left(\mathbf{W}_2\mathbf{W}_1\tilde{\mathbf{\Sigma}}_i^{yx^T}\right) + \frac{1}{2}\operatorname{Tr}\left(\tilde{\mathbf{\Sigma}}_i^{yy}\right) \\
&= \frac{1}{2}\operatorname{Tr}\left(\left(\mathbf{W}_2\mathbf{W}_1 - \tilde{\mathbf{\Sigma}}_i^{yx}\right)\left(\mathbf{W}_2\mathbf{W}_1 - \tilde{\mathbf{\Sigma}}_i^{yx}\right)^T - \tilde{\mathbf{\Sigma}}_i^{yx}\tilde{\mathbf{\Sigma}}_i^{yx^T}\right) + \frac{1}{2}\left(\tilde{\mathbf{\Sigma}}_i^{yy}\right) \\
&= \frac{1}{2}\left\|\mathbf{W}_2\mathbf{W}_1 - \tilde{\mathbf{\Sigma}}_i^{yx}\right\|_F^2 \underbrace{- \frac{1}{2}\operatorname{Tr}\left(\tilde{\mathbf{\Sigma}}_i^{yx}\tilde{\mathbf{\Sigma}}_i^{yx^T}\right) + \frac{1}{2}\left(\tilde{\mathbf{\Sigma}}_i^{yy}\right)}_{c}. \qquad \text{(A.257)}
\end{aligned}
$$

Hence, the extent of forgetting, denoted as $\mathcal{F}_i$ for task $\mathcal{T}_i$ during training on task $\mathcal{T}_k$ subsequent to training the network on task $\mathcal{T}_j$, specifically, the relative change in loss, is entirely dictated by the similarity structure among tasks.

$$
\begin{aligned}
\mathcal{F}_i\left(\mathcal{T}_j, \mathcal{T}_k\right) &= \mathcal{L}_i\left(\mathcal{T}_k\right) - \mathcal{L}_i\left(\mathcal{T}_j\right) \\
&= \frac{1}{2}\left\|\tilde{\mathbf{\Sigma}}_k^{yx} - \tilde{\mathbf{\Sigma}}_i^{yx}\right\|_F^2 + c - \frac{1}{2}\left\|\mathbf{W}_2\mathbf{W}_1 - \tilde{\mathbf{\Sigma}}_i^{yx}\right\|_F^2 - c \\
&= \frac{1}{2}\left(\left\|\tilde{\mathbf{\Sigma}}_k^{yx} - \tilde{\mathbf{\Sigma}}_i^{yx}\right\|_F^2 - \left\|\mathbf{W}_2\mathbf{W}_1 - \tilde{\mathbf{\Sigma}}_i^{yx}\right\|_F^2\right). \qquad \text{(A.258)}
\end{aligned}
$$

It is important to note that the amount of forgetting is a function of the weight trajectories. Therefore, we have analytical solutions for trajectories of forgetting.

A.4.2 Reversal learning

In the following discussion, we assume that the input and output dimensions are equal. We denote the i-th columns of the left and right singular vectors as $\mathbf{u}_i$, $\tilde{\mathbf{u}}_i$, and $\mathbf{v}_i$, $\tilde{\mathbf{v}}_i$, respectively. Reversal learning occurs when both the task and the initial network function share the same left and right singular vectors, i.e., $\mathbf{U} = \tilde{\mathbf{U}}$ and $\mathbf{V} = \tilde{\mathbf{V}}$, with the exception of one or more columns of the left singular vectors, where the direction is reversed: $-\mathbf{u}_i = \tilde{\mathbf{u}}_i$. It is important to note that if a reversal occurs in the right singular vectors, such that $-\mathbf{v}_i = \tilde{\mathbf{v}}_i$, this can be equivalently represented as a reversal in the left singular vectors, as the signs of the right and left singular vectors are interchangeable.

In the reversal learning setting, both $\mathbf{B} = \mathbf{S}_2\tilde{\mathbf{U}}^T\tilde{\mathbf{U}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}}) + \mathbf{S}_1\mathbf{V}^T\tilde{\mathbf{V}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})$ and $\mathbf{C} = \mathbf{S}_2\tilde{\mathbf{U}}^T\tilde{\mathbf{U}}(\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}}) - \mathbf{S}_1\mathbf{V}^T\tilde{\mathbf{V}}(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})$ are diagonal matrices.

In the case where lambda is zero the diagonal entries of $\mathbf{C}$ are zero if the singular vectors are aligned and non zero if they are reversed. Similarly, diagonal entries of $\mathbf{B}$ are non-zero if the singular vectors are aligned and zero if they are reversed. Therefore, in the case of reversal learning, $\mathbf{B}$ is a diagonal matrix with 0 values and thus is not invertible. As a consequence, the learning dynamics cannot be described by Eq. A.101. However, as $\mathbf{B}$ and $\mathbf{C}$ are diagonal matrices, the learning dynamics simplify. Let $\mathbf{b}_i$, $\mathbf{c}_i$, $\mathbf{s}_i$ and $\tilde{\mathbf{s}}_i$ denote the i-th diagonal entry of $\mathbf{B}$, $\mathbf{C}$, $\mathbf{S}$ and $\tilde{\mathbf{S}}$ respectively, then the network dynamics can be rewritten as

$$\begin{aligned}
\mathbf{W}_2\mathbf{W}_1(t) &= \frac{1}{2}\tilde{\mathbf{U}}\left[(\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{B}^T + (\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{C}^T\right) \\
&\quad \left[\mathbf{S}_\lambda^{-1} + \frac{1}{4}\mathbf{B}\left(e^{2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}} - \mathbf{I}\right)\tilde{\mathbf{S}}_\lambda^{-1}\mathbf{B}^T - \frac{1}{4}\mathbf{C}\left(e^{-2\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}} - \mathbf{I}\right)\tilde{\mathbf{S}}_\lambda^{-1}\mathbf{C}^T\right]^{-1} \\
&\quad \frac{1}{2}\left((\tilde{\mathbf{G}}-\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{B} - (\tilde{\mathbf{G}}+\tilde{\mathbf{H}}\tilde{\mathbf{G}})e^{-\tilde{\mathbf{S}}_\lambda\frac{t}{\tau}}\mathbf{C}\right)\tilde{\mathbf{V}}^T \\
&= \sum_{i=1}^{N_i} \frac{\mathbf{b}_i^2 e^{2\tilde{\mathbf{s}}_{\lambda i}\frac{t}{\tau}} - \mathbf{c}_i^2 e^{-2\tilde{\mathbf{s}}_{\lambda i}\frac{t}{\tau}}}{4{\mathbf{s}_\lambda}_i^{-1} + \mathbf{b}_i^2 e^{2\tilde{\mathbf{s}}_{\lambda i}\frac{t}{\tau}}\tilde{\mathbf{s}}_{\lambda\mathbf{i}}^{-1} - \mathbf{b}_i^2\tilde{\mathbf{s}}_{\lambda\mathbf{i}}^{-1} - \mathbf{c}_i^2 e^{-2\tilde{\mathbf{s}}_{\lambda i}\frac{t}{\tau}}\tilde{\mathbf{s}}_{\lambda\mathbf{i}}^{-1} + \mathbf{c}_i^2\tilde{\mathbf{s}}_{\lambda\mathbf{i}}^{-1}}\tilde{\mathbf{u}}_i\tilde{\mathbf{v}}_i^T \\
&= \sum_{i=1}^{N_i} \frac{{\mathbf{s}_\lambda}_i\mathbf{b}_i^2\tilde{\mathbf{s}}_{\lambda\mathbf{i}} - {\mathbf{s}_\lambda}_i\mathbf{c}_i^2\tilde{\mathbf{s}}_i e^{-4\tilde{\mathbf{s}}_i\frac{t}{\tau}}}{4\tilde{\mathbf{s}}_{\lambda\mathbf{i}}e^{-2\tilde{\mathbf{s}}_i\frac{t}{\tau}} + {\mathbf{s}_\lambda}_i\mathbf{b}_i^2\left(1 - e^{-2\tilde{\mathbf{s}}_{\lambda i}\frac{t}{\tau}}\right) + {\mathbf{s}_\lambda}_i\mathbf{c}_i^2\left(e^{-2\tilde{\mathbf{s}}_{\lambda i}\frac{t}{\tau}} - e^{-4\tilde{\mathbf{s}}_{\lambda i}\frac{t}{\tau}}\right)}\tilde{\mathbf{u}}_i\tilde{\mathbf{v}}_i^T.
\end{aligned} \tag{A.259}$$

It follows, that in the reversal learning case, i.e. $\mathbf{b} = 0$, for each reversed singular vector, the dynamics vanish to zero

$$\lim_{t\to\infty} \frac{-{\mathbf{s}_\lambda}_i\mathbf{c}_i^2\tilde{\mathbf{s}}_i e^{-4\tilde{\mathbf{s}}_{\lambda i}\frac{t}{\tau}}}{4\tilde{\mathbf{s}}_{\lambda,\mathbf{i}}e^{-2\tilde{\mathbf{s}}_{\lambda i}\frac{t}{\tau}} + \mathbf{s}_i\mathbf{c}_i^2\left(e^{-2\tilde{\mathbf{s}}_{\lambda i}\frac{t}{\tau}} - e^{-4\tilde{\mathbf{s}}_{\lambda i}\frac{t}{\tau}}\right)}\tilde{\mathbf{u}}_i\tilde{\mathbf{v}}_i^T = 0. \tag{A.260}$$

Analytically, the learning dynamics are initialised on and remain along the separatrix of a saddle point until the corresponding singular value of the network function decreases to zero and stays there, indicating convergence to the saddle point. In numerical simulations, however, the learning dynamics can escape the saddle points due to the imprecision of floating-point arithmetic. Despite this, numerical optimisation still experiences significant delays, as escaping the saddle point is time-consuming [168]. In contrast, when the singular vectors are aligned ($\mathbf{c} = 0$), the equation governing temporal dynamics, as described in Saxe et al. [243], is recovered. Under these conditions, training succeeds, with the singular value of the network function converging to its target value.

$$\begin{aligned}\lim_{t\to\infty}\sum_{i=1}^{N_i}\frac{\mathbf{s}_{\lambda\mathbf{i}}\mathbf{b}_i^2\tilde{\mathbf{s}}_{\lambda\mathbf{i}}}{4\tilde{\mathbf{s}}_{\lambda\mathbf{i}}e^{-2\tilde{\mathbf{s}}_{\lambda i}\frac{t}{\tau}}+\mathbf{s}_{\lambda\mathbf{i}}\mathbf{b}_i^2\left(1-e^{-2\tilde{\mathbf{s}}_{\lambda i}\frac{t}{\tau}}\right)}\tilde{\mathbf{u}}_i\tilde{\mathbf{v}}_i^T &= \frac{\mathbf{s}_{\lambda\mathbf{i}}\mathbf{b}_i^2\tilde{\mathbf{s}}_{\lambda\mathbf{i}}}{\mathbf{s}_{\lambda\mathbf{i}}\mathbf{b}_i^2}\tilde{\mathbf{u}}_i\tilde{\mathbf{v}}_i^T \\ &= \tilde{\mathbf{s}}_{\lambda\mathbf{i}}\tilde{\mathbf{u}}_i\tilde{\mathbf{v}}_i^T. \qquad (A.261)\end{aligned}$$

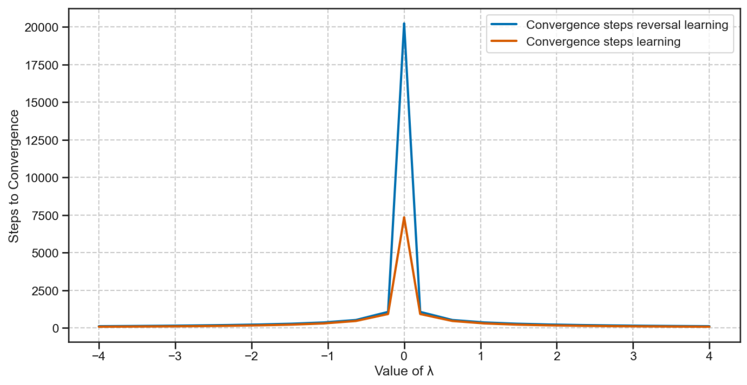


Figure A.9: Plot showing the steps to convergence for two tasks: (1) the reversal learning task and (2) a randomly sampled continual learning task across a range of λ values. The reversal learning task exhibits catastrophic slowing at $\lambda = 0$.

In the case where λ is non-zero, the diagonal of $\mathbf{C}$ are also non-zero; this is true regardless of whether they are reversed or aligned. Similarly, the diagonal entries of $\mathbf{B}$ remain non-zero whether the singular vectors are aligned or reversed. Therefore, when λ is non-zero reversal learning always succeeds as shown in Fig. A.9

A.4.2.1 Exact learning dynamics in shallow networks

To provide a point of comparison to our deep linear network results, here we derive a solution for the temporal dynamics of reversal learning in a shallow network. The network's weights are optimised using full batch gradient descent with learning rate η (or equivalently time constant $\tau = 1/\eta$) on the mean squared error loss given in Eq. 3.2, yielding the first task dynamics

$$\tau \frac{d}{dt}\mathbf{W} = \tilde{\boldsymbol{\Sigma}}^{yx} - \mathbf{W}\tilde{\boldsymbol{\Sigma}}^{xx}, \tag{A.262}$$

where $\tilde{\boldsymbol{\Sigma}}^{xx}$ and $\tilde{\boldsymbol{\Sigma}}^{yx}$ is the input and input-output correlation matrices of the dataset. We define

$$\mathrm{SVD}(\mathbf{W}(0)) = \mathbf{U}\mathbf{S}\mathbf{V}^T \quad \text{and} \quad \mathrm{SVD}(\tilde{\boldsymbol{\Sigma}}^{yx}) = \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T. \tag{A.263}$$

motivating the change of variable $\mathbf{W} = \mathbf{U}\overline{\mathbf{W}}\mathbf{V}^T$. We project the weight into the basis of the initialisation

$$\tau \frac{d}{dt}\mathbf{U}\overline{\mathbf{W}}\mathbf{V}^T = \tilde{\boldsymbol{\Sigma}}^{yx} - \mathbf{U}\overline{\mathbf{W}}\mathbf{V}^T\tilde{\boldsymbol{\Sigma}}^{xx}, \tag{A.264}$$

$$\tau \frac{d}{dt}\mathbf{U}\overline{\mathbf{W}}\mathbf{V}^T = \mathbf{U}\mathbf{U}^T\tilde{\boldsymbol{\Sigma}}^{yx}\mathbf{V}\mathbf{V}^T - \mathbf{U}\overline{\mathbf{W}}\mathbf{V}^T\tilde{\boldsymbol{\Sigma}}^{xx}, \tag{A.265}$$

$$\tau \frac{d}{dt}\overline{\mathbf{W}} = \mathbf{U}^T\tilde{\boldsymbol{\Sigma}}^{yx}\mathbf{V} - \overline{\mathbf{W}}\tilde{\boldsymbol{\Sigma}}^{xx}. \tag{A.266}$$

Under the assumption of whitened inputs (Assumption 1), the dynamics yields

$$\tau \frac{d}{dt}\overline{\mathbf{W}} = \mathbf{U}^T\tilde{\boldsymbol{\Sigma}}^{yx}\mathbf{V} - \overline{\mathbf{W}}. \tag{A.267}$$

Defining $\overline{\mathbf{W}}_{ii} = b_i$ the diagonal element of the matrix, encoding the strength of the mode i transmitted by the input-to-output weight. Similarly, we write $(\mathbf{U}^T\tilde{\boldsymbol{\Sigma}}^{yx}\mathbf{V})_{ii} = k_i$. Assuming decoupled initial conditions, we obtain the scalar dynamics

$$\tau \frac{d}{dt}b_i = k_i - b_i \tag{A.268}$$

with solution

$$b_i = k_i(1 - e^{\frac{-t}{\tau}}) + b_i^0 e^{\frac{-t}{\tau}}. \tag{A.269}$$

Reverting the change of variable, the weight trajectory yields

$$\mathbf{W} = \mathbf{U}\mathbf{B}(t)\mathbf{V}^T. \tag{A.270}$$

This solution is very similar to the one proposed by Saxe et al. [247]. However, the key here is that k_i can have negative values. k_i is negative whenever a vector is in the opposite direction to the initialisation (as in the reversal learning setting). We show in Fig. 3.12 that the analytical solution derived above matches the numerical temporal dynamics. From Eq. A.269, we note that the shallow network cannot display catastrophic slowing.

A.4.3 Transfer learning

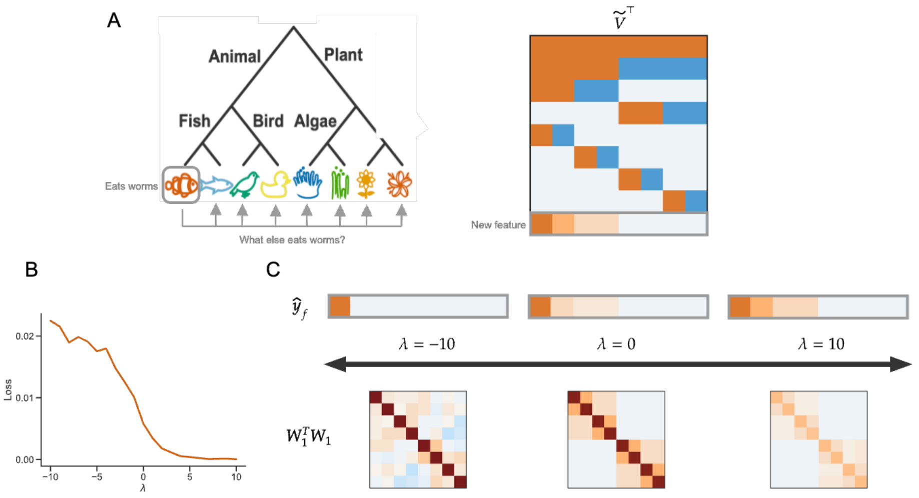


Figure A.10: Transfer learning for different λ. **A** A new feature (such as 'eats worms') is introduced to the dataset after training on the hierarchical semantic learning task (Sec. 3.4.2). A randomly initialised row is added to $\mathbf{W}_2$ and trained on a single item with the new feature (for example, the goldfish), with the rest of the network frozen. The network is then tested on the transfer of the new feature to other items, such that items closer to the goldfish in the hierarchy are more likely to have the same feature. **B** The generalisation loss on the untrained items with the new feature decreases as λ increases. **C** As λ increases positively, networks better transfer the hierarchical structure of the data to the representation of the new feature.

We study how the representations learned for different λ initialisations impact generalisation of properties of the data. To do this, we consider the case where a new feature is associated to a learned item in a dataset and how this new feature may then be related to other items based on prior knowledge. In particular, we first train each network (for different values of $-10 \leq \lambda \leq 10$) on the hierarchical semantic learning task in Sec. 3.4.2 and then add a new feature (e.g., 'eats worms') to a single item (e.g., the goldfish) (Fig. A.10A), correspondingly increasing the output dimension to represent the novel feature. In order to learn the new feature without affecting prior knowledge, we append a randomly initialised row to $\mathbf{W}_2$ and train it on the single item with the new feature, while keeping the rest of the network frozen. Thus, we only change the weights from the hidden layer to the new feature which may produce different behaviour depending on how the hidden layer representations vary based on λ. After training on the new feature-item association, we query the network with the rest of the data to observe how the new feature is associated with the other items. We find that as λ increases positively, the network better transfers the hierarchy such that it projects the feature onto items based on their distance to the trained item (Fig. A.10B,C). For example, after learning that a goldfish eats worms, the network can extrapolate the hierarchy to infer that another fish, or birds, may also eat worms; instead, plants are not likely to eat worms. Alternatively, as λ becomes more negative, the network ceases to infer any hierarchical structure and only learns to map the new feature to the single item trained on. In this case, after learning that a goldfish eats worms, the network does not infer that other fish, birds, or plants may also eat worms.

Interestingly, this setting highlights how asymmetries in the representations yielded by different λ can actually benefit transfer and generalisation. This can be shown by observing that the learning of a new feature association only depends on the first layer $\mathbf{W}_1$. Let $\hat{\mathbf{y}}_f$ denote the vector of the representation of the new feature f across items i in the dataset. Additionally, let $\mathbf{W}_2^{(f)T}$ be the new row of weights appended to $\mathbf{W}_2$ which map the hidden layer to the new feature. Following Saxe et al. [247], if $\mathbf{w}_2^{(f)T}$ is initialised with small random weights and trained on item $\tilde{\mathbf{H}}_i$, it will converge to

$$\mathbf{w}_2^{(f)T} = \tilde{\mathbf{H}}_i^T \mathbf{W}_1^T / ||\mathbf{W}_1 \tilde{\mathbf{H}}_i||_2^2 \tag{A.271}$$

$$\hat{\mathbf{y}}_f = (\tilde{\mathbf{H}}_i^T \mathbf{W}_1^T \mathbf{W}_1 \tilde{\mathbf{H}}) / ||\mathbf{W}_1 \tilde{\mathbf{H}}_i||_2^2 \tag{A.272}$$

From this we can see that differences in the representations of the new feature across items $\hat{\mathbf{y}}_f$ across λ are only influenced by $\mathbf{W}_1$. In the case of the rich learning regime where $\lambda = 0$, the semantic relationship between features and items is distributed across both layers. Instead, when $\lambda > 0$, the second layer $\mathbf{W}_2$ exhibits *lazy* learning, yielding an output representation $\mathbf{W}_2 \mathbf{W}_2^T$ of a weighted identity matrix. However, the first layer $\mathbf{W}_1$ still learns a *rich* representation of the hierarchy, albeit at a smaller scaling. Furthermore in the $\lambda > 0$ case, learning of the hierarchy occurs in the first layer, allowing it to more readily transfer this structure to the learning of a new feature (which only depends on the first layer). Thus, in this case, the 'shallowing' of the network into the first layer is actually beneficial. Finally, we can also observe the opposite case when $\lambda < 0$. Here, *rich* learning happens in the second layer, while the first layer is *lazy* and learns to represent a weighted identity matrix. As such, these networks do not learn to transfer the hierarchy of different items to the new feature.

A.4.4 Fine-tuning

It is a common practice to pre-train neural networks on a large auxiliary task before fine-tuning them on a downstream task with limited samples. Despite the widespread use of this approach, the dynamics and outcomes of this method remain poorly understood. In our study, we provide a theoretical foundation for the empirical success of fine-tuning, aiming to improve our understanding of how performance depends on the initialisation. We're interested in understanding how changing the λ-balancedness after pre-training may impact fine-tuning on a new dataset. We use λ_{PT} to denote how networks are first initialised prior to pretraining, and λ_{FT} to how they are *re-balanced* after pre-training and before fine-tuning on a new task. Similar to the previous section, we first train each network (for different values of $-10 \leq \lambda_{PT} \leq 10$) on the hierarchical semantic learning task. We then change

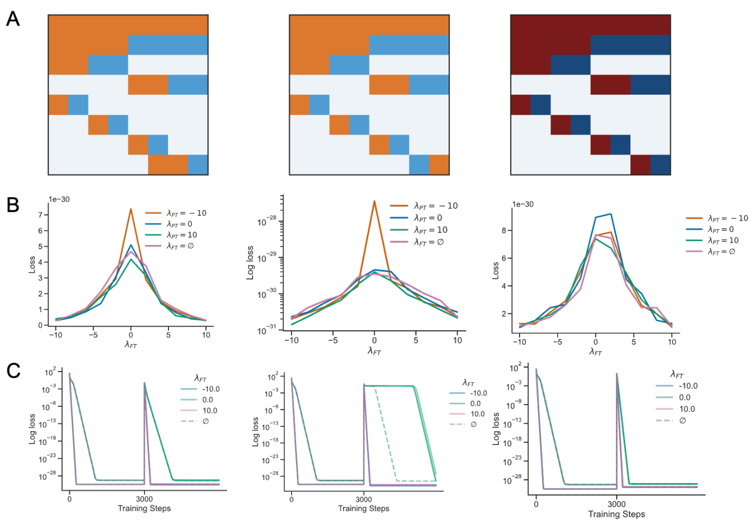


Figure A.11: Fine-tuning performance on three tasks for different re-balancing λ_{FT}. **A** After training on the hierarchical semantic learning task (Sec. 3.4.2), networks are re-balanced and trained on one of three tasks: adding an existing feature from one item to another item in the hierarchy (*left*), the reversal learning task in Appendix A.4.2 (*center*), or a scaled version of the hierarchy where each singular value is scaled by 2 (*right*). **B** Change in loss on the new task across different λ_{FT} for different λ_{PT}. As λ_{FT} approaches 0, the loss on the new task increases across all λ_{PT}. Interestingly, networks that are not re-balanced prior to fine-tuning ($\lambda_{FT} = \emptyset$) perform similarly to networks that are re-balanced to the same values ($\lambda_{FT} = \lambda_{PT}$). **C** Dynamics of the loss across the first pre-training task and the new fine-tuning task. Networks re-balanced to $\lambda_{FT} = 0$ consistently learn slower across all tasks compared to networks that are re-balanced to larger magnitude values ($|\lambda_{FT}| > 0$)

the λ-balancedness of each network (for different values of $-10 \leq \lambda_{FT} \leq 10$) and retrain on a new dataset to observe how this impacts fine-tuning for different values and compare to networks that are not re-balanced to some λ_{FT} ($\lambda_{FT} = \emptyset$) after initial pre-training. In particular, to reset the λ-balancedness of a pre-trained network to λ_{FT}, we rescale the singular values of each layer ($\mathbf{S}_1, \mathbf{S}_2$) using the singular values of the entire network function ($\mathbf{S} = \mathbf{U}^T \mathbf{W}_2 \mathbf{W}_1 \mathbf{V}$), while keeping the left and right singular vectors of the network unchanged.

We consider three different tasks to fine-tune the networks on. In the first, we add an existing feature from one item to another item in the hierarchy in order to disrupt the structure of the left and right singular vectors. In the second task, we consider the same reversal learning task discussed in Appendix A.4.2, where one column of the right singular vectors are reversed such that $-\boldsymbol{v}_i = \tilde{\boldsymbol{v}}_i$. Finally, we consider a scaled version of the hierarchy where each singular value is scaled by 2.

Across all the tasks we consider, we consistently find that fine-tuning performance improves as networks are re-balanced to larger values of λ_{FT} and, conversely, decreases as λ_{FT} approaches 0. Networks re-balanced to $\lambda_{FT} = 0$ also learn more slowly compared to $|\lambda_{FT}| > 0$. Interestingly, when studying networks that are *not* re-balanced prior to fine-tuning ($\lambda_{FT} = \emptyset$; but *are* first initialised prior to pretraining to λ_{PT}), we see that they perform similarly on the new tasks to networks that are re-balanced to $\lambda_{FT} = \lambda_{PT}$.

In this work, we derive the precise dynamics of two-layer linear. While straightforward in design, these architectures are foundational in numerous machine learning applications, particularly in the implementation of Low Rank Adapters (LoRA)[124]. A key innovation in LoRA is to parameterise the update of a large weight matrix $\mathbf{W} \in \mathbb{R}^{d\times d}$ within a language model as $\Delta\mathbf{W} = \mathbf{AB}$, the product of two low-rank matrices $\mathbf{A} \in \mathbb{R}^{d\times r}$ and $\mathbf{B} \in \mathbb{R}^{r\times d}$, where only $\mathbf{A}$ and $\mathbf{B}$ are trained. To ensure $\Delta\mathbf{W} = 0$ at initialisation, it is standard practice to initialise $\mathbf{A} \sim \mathcal{N}(0, \sigma^2)$ and $\mathbf{B} = 0$ ([124, 117]. It is noteworthy that this parameterisation, $\Delta\mathbf{W} = \mathbf{AB}$, effectively embeds a two-layer linear network within the language model. When $r \ll d$, this initialisation scheme approximately adheres to our λ-balanced condition, with σ^2 playing the role of the balance parameter λ. Investigating how the initialisation scale of $\mathbf{A}$ and $\mathbf{B}$ influences fine-tuning dynamics under LoRA, and connecting this to our work on λ-balanced two-layer linear networks and their role in feature learning, represents an intriguing avenue for future exploration. This perspective aligns with recent studies suggesting that low-rank fine-tuning operates in a “lazy” regime, as well as work examining how the initialisation of $\mathbf{A}$ or $\mathbf{B}$ affects fine-tuning performance [183, 117]. Our framework offers a potential bridge to understanding these phenomena more comprehensively. While a detailed exploration of fine-tuning performance lies beyond the scope of this work, it remains an important direction for future research.

A.4.5 Revising structured knowledge

We assume that the network learn a first task with input-output correlation $\mathbf{\Sigma} = \mathbf{USV}$. We explore the dynamics of the deep network when learning a subsequent task with input-output correlations $\tilde{\mathbf{\Sigma}} = \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}$. Specifically, we show that the gradient step increment is zero for the dimension of the eigen-basis when the two tasks are schema consistent. We have

$$\tau\frac{d}{dt}\mathbf{W}_1 = -\mathbf{W}_2^T(\tilde{\mathbf{\Sigma}}^{xy} - \mathbf{W}_2\mathbf{W}_1\tilde{\mathbf{\Sigma}}^{x}) \tag{A.273}$$

$$\tau\frac{d}{dt}\mathbf{W}_2 = -(\tilde{\mathbf{\Sigma}}^{yx} - \mathbf{W}_2\mathbf{W}_1\tilde{\mathbf{\Sigma}}^{x})\mathbf{W}_1^T \tag{A.274}$$

Assume whitened inputs for the second task (ie. $\tilde{\mathbf{\Sigma}}^{x} = \mathbf{I}$) and substituting $\tilde{\mathbf{\Sigma}}^{xy} = \tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T$, we can rewrite Eq. A.274 and Eq. A.273 as

$$\tau\frac{d}{dt}\mathbf{W}_1 = -\mathbf{W}_2^T(\tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T - \mathbf{W}_2\mathbf{W}_1), \tag{A.275}$$

$$\tau\frac{d}{dt}\mathbf{W}_2 = -(\tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T - \mathbf{W}_2\mathbf{W}_1)\mathbf{W}_1^T. \tag{A.276}$$

We consider the first epoch of the second task. Therefore, we can rewrite Eq. A.276 and Eq. A.275 using the eigen-representation of the weights $\mathbf{W}_1, \mathbf{W}_2$ at the end of the first task

$$\tau\frac{d}{dt}\mathbf{W}_1 = -\mathbf{R}\sqrt{\mathbf{S}}\mathbf{U}^T(\tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T - \mathbf{U}\sqrt{\mathbf{S}}\mathbf{R}^T\mathbf{R}\sqrt{\mathbf{S}}\mathbf{V}^T), \tag{A.277}$$

$$\tau\frac{d}{dt}\mathbf{W}_2 = -(\tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T - \mathbf{U}\sqrt{\mathbf{S}}\mathbf{R}^T\mathbf{R}\sqrt{\mathbf{S}}\mathbf{V}^T)\mathbf{V}\sqrt{\mathbf{S}}\mathbf{R}^T, \tag{A.278}$$

which simplifies to

$$\tau\frac{d}{dt}\mathbf{W}_1 = -(\mathbf{R}\sqrt{\mathbf{S}}\mathbf{U}^T(\tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T - \mathbf{USV}^T)), \tag{A.279}$$

$$\tau\frac{d}{dt}\mathbf{W}_2 = -(\tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T - \mathbf{USV}^T)\mathbf{V}\sqrt{\mathbf{S}}\mathbf{R}^T. \tag{A.280}$$

We consider the change variables to $\overline{\mathbf{W}}_1$ and $\overline{\mathbf{W}}_2$ where

$$\mathbf{W}_1 = \mathbf{R}\overline{\mathbf{W}}_1\mathbf{V}^T, \tag{A.281}$$

$$\mathbf{W}_2 = \mathbf{U}\overline{\mathbf{W}}_2\mathbf{R}^T, \tag{A.282}$$

giving

$$\tau\frac{d}{dt}\overline{\mathbf{W}}_1 = -(\sqrt{\mathbf{S}}\mathbf{U}^T(\tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T\mathbf{V} - \mathbf{U}\mathbf{S})), \tag{A.283}$$

$$\tau\frac{d}{dt}\overline{\mathbf{W}}_2 = -(\tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T - \mathbf{U}\mathbf{S}\mathbf{V}^T)\mathbf{V}\sqrt{\mathbf{S}}. \tag{A.284}$$

Further simplifiying to

$$\tau\frac{d}{dt}\overline{\mathbf{W}}_1 = -(\sqrt{\mathbf{S}}(\mathbf{U}^T\tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T\mathbf{V} - \mathbf{S})), \tag{A.285}$$

$$\tau\frac{d}{dt}\overline{\mathbf{W}}_2 = -(\mathbf{U}^T\tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T\mathbf{V} - \mathbf{S})\sqrt{\mathbf{S}}. \tag{A.286}$$

The differential equation is equal to zero when

$$(\mathbf{U}^T\tilde{\mathbf{U}}\tilde{\mathbf{S}}\tilde{\mathbf{V}}^T\mathbf{V} - \mathbf{S}) = 0. \tag{A.287}$$

We first observe that the equation above is satisfied by modes that share the same eigenvectors and eigenvalues. If the modes of both $\mathbf{W}_1$ and $\mathbf{W}_2$ remain unchanged in the initial step—that is, if their respective gradient updates are zero—the inequality will continue to hold in all subsequent steps. However, when the input–output correlations of the first and second tasks have different bases in their singular value decompositions, we expect modifications to both eigenvalues and eigenvectors. Consequently, replay can only reduce learning time when the two tasks share the same eigenstructure, thereby motivating the replay algorithm.

A.5 Implementation and simulations

The details of the simulation studies are described as follows. Specifically, N_i, N_h, and N_o represent the dimensions of the input, hidden layer, and output (target), respectively.

A.5.1 Lambda-balanced weight initialisation

In practice, to initialise the network with lambda-balanced weights, we use Algorithm 1.

A.5.2 Tasks

In the following, we describe the different tasks that are used throughout the simulation studies.

A.5.2.1 Random regression task

In the random regression task, the inputs $\mathbf{X} \in \mathbb{R}^{N_i \times N}$ are generated from a standard normal distribution, $\mathbf{X} \sim \mathcal{N}(\mu = 0, \sigma = 1)$. The input data $\mathbf{X}$ is then whitened to satisfy $\frac{1}{N}\mathbf{X}\mathbf{X}^T = \mathbf{I}$. The target values $\mathbf{Y} \in \mathbb{R}^{N_o \times N}$ are independently sampled from a normal distribution with variance scaled according to the number of output nodes, $\mathbf{Y} \sim \mathcal{N}(\mu = 0, \alpha = \frac{1}{\sqrt{N_o}})$. Consequently, the network inputs and target values are uncorrelated Gaussian noise, implying that a linear solution may not always exist.

A.5.2.2 Semantic hierarchy

We use the same task as in Saxe et al. [243] and modify it to match the theoretical dynamics. The modification ensures that the inputs are whitened. In the semantic hierarchy task, input items are represented as one-hot vectors, i.e., $\mathbf{X} = \frac{\mathbf{I}}{8}$. The corresponding target vectors, $\mathbf{y}_i$, encode the item's position within the hierarchical tree. Specifically, a value of 1 indicates that the item is a left child of a node, -1 denotes a right child, and 0 indicates that the item is not a child of that node. For example, consider the blue fish: it is a blue fish, a left child of the root node, a left child of the animal node, not part of the plant branch, a right child of the fish node, and not part of the bird, algae, or flower branches, resulting in the label $[1,1,1,0,-1,0,0,0]$. The labels for all objects in the semantic tree, as shown in Fig. 3.4 A, are given by:

Algorithm 1 Get λ-*balanced*

```
function GET_LAMBDA_BALANCED($\lambda$, $in\_dim$, $hidden\_dim$, $out\_dim$, $\sigma = 1$)
    if $out\_dim > in\_dim$ and $\lambda < 0$ then
        raise Exception('Lambda must be positive if out_dim ¿ in_dim')
    end if
    if $in\_dim > out\_dim$ and $\lambda > 0$ then
        raise Exception('Lambda must be positive if in_dim ¿ out_dim')
    end if
    if $hidden\_dim < \min(in\_dim, out\_dim)$ then
        raise Exception('Network cannot be bottlenecked')
    end if
    if $hidden\_dim > \max(in\_dim, out\_dim)$ and $\lambda \neq 0$ then
        raise Exception('hidden_dim cannot be the largest dimension if lambda is not 0')
    end if
    $\mathbf{W}_1 \leftarrow \sigma \cdot \text{random normal matrix}(hidden\_dim, in\_dim)$
    $W_2 \leftarrow \sigma \cdot \text{random normal matrix}(out\_dim, hidden\_dim)$
    $[U, S, Vt] \leftarrow \text{SVD}(W_2 \cdot \mathbf{W}_1)$
    $R \leftarrow \text{random orthonormal matrix}(hidden\_dim)$
    $S2_{equal\_dim} \leftarrow \sqrt{\left(\sqrt{\lambda^2 + 4 \cdot S^2} + \lambda\right)/2}$
    $S1_{equal\_dim} \leftarrow \sqrt{\left(\sqrt{\lambda^2 + 4 \cdot S^2} - \lambda\right)/2}$
    if $out\_dim > in\_dim$ then
        $S2 \leftarrow \begin{bmatrix} S2_{equal\_dim} & 0 \\ 0 & 0_{hidden\_dim - in\_dim} \end{bmatrix}$
        $S1 \leftarrow \begin{bmatrix} S1_{equal\_dim} \\ 0 \end{bmatrix}$
    else if $in\_dim > out\_dim$ then
        $S1 \leftarrow \begin{bmatrix} S1_{equal\_dim} & 0 \\ 0 & 0_{hidden\_dim - out\_dim} \end{bmatrix}$
        $S2 \leftarrow \begin{bmatrix} S2_{equal\_dim} & 0 \end{bmatrix}$
    end if
    $init\_W_2 \leftarrow U \cdot S2 \cdot R^T$
    $init\_\mathbf{W}_1 \leftarrow R \cdot S1 \cdot Vt$
    return $(init\_\mathbf{W}_1, init\_W_2)$
end function
```

$$\mathbf{Y} = 8 * \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & 1 & -1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & -1 & -1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix}. \tag{A.288}$$

The singular value decomposition (SVD) of the corresponding correlation matrix, $\tilde{\mathbf{\Sigma}}^{yx}$, is not unique due to identical singular values: the first two, the third and fourth, and the last four values are the same. To align the numerical and analytical solutions, this permutation invariance is addressed by adding a small perturbation to each column $\mathbf{y}_i$, for $i \in 1, ..., N$, of the labels:

$$\mathbf{y}_i = \mathbf{y}_i \cdot \left(1 + \frac{0.1}{i}\right), \tag{A.289}$$

resulting in singular values that are nearly, but not exactly, identical.

A.5.3 Figure 3.1

Panels B illustrates three simulations conducted on the same task with varying initial λ-balanced weights respectively $\lambda = -2$, $\lambda = 0$, $\lambda = 2$. The regression task parameters were set with $(\sigma = \sqrt{10})$. The network architecture consisted of $N_i = 3$, $N_h = 2$, $N_o = 2$,with a learning rate of $\eta = 0.0002$. The batch size is $N = 10$. The zero-balanced weights are initialised with variance $\sigma = 0.00001$. The lambda-balanced network are initialised with $\sigma xy = \sqrt{1}$ of a random regression task with same architecture.

On Panel C , we plot the balancedness $\mathbf{W}_2(0)^T\mathbf{W}_2(0) - \mathbf{W}_1(0)\mathbf{W}_1(0)^T$ for a two layer network initialised with LeCun initialisation with dimension $N_i = 40$,N_h=

120 ,N_o=250

A.5.4 Figure 3.2

Panel A, B, C illustrates three simulations conducted on the same task with varying initial λ-balanced weights respectively $\lambda = -2$, $\lambda = 0$, $\lambda = 2$ according to the initialisation scheme described in Appendix A.5. The regression task parameters were set with $(\sigma = \sqrt{10})$. The network architecture consisted of $N_i = 3$, $N_h = 2$, $N_o = 2$ with a learning rate of $\eta = 0.0002$. The batch size is $N = 10$. The zero-balanced weights are initialised with variance $\sigma = 0.00001$. The lambda-balanced network are initialised with $sigmaxy = \sqrt{1}$ of a random regression task with same architecture.

A.5.5 Figure 3.3

Panel A, B, C illustrates three simulations conducted on the same task with varying initial λ-balanced weights respectively $\lambda = -2$, $\lambda = 0$, $\lambda = 2$ according to the initialisation scheme described in Appendix A.5. The regression task parameters were set with $(\sigma = \sqrt{12})$. The network architecture consisted of $N_i = 3$, $N_h = 3$, $N_o = 3$ with a learning rate of $\eta = 0.0002$. The batch size is $N = 5$. The zero-balanced weights are initialised with variance $\sigma = 0.0009$. The lambda-balanced network are initialised with $\Sigma xy = \sqrt{12}$ of a random regression task with same architecture.

A.5.6 Figure 3.4

In Panel A presents a semantic learning task with the SVD of the input-output correlation matrix of the task. **U** and **V** represent the singular vectors, and **S** contains the singular values. This decomposition allows us to compute the respective RSMs as $\mathbf{USU}^\top$ for the input and $\mathbf{VSV}^\top$ for the output task. The rows and columns in the SVD and RSMs are ordered identically to the items in the hierarchical tree.

The results in Panel B display simulation outcomes, while Panel C presents theoretical input and output representation matrices at convergence for a network trained on the semantic task described in [243],. These matrices are generated using varying initial λ-balanced weights set at $\lambda = -2$, $\lambda = 0$, and $\lambda = 2$, following the initialisation scheme outlined in Appendix A.5. The network architecture includes $N_i = 8$, $N_h = 8$, and $N_o = 8$ with a learning rate of $\eta = 0.001$ and a batch size of $N = 8$. Zero-balanced weights are initialised with a variance of $\sigma = 0.00001$, while λ-balanced networks are initialised with $\sigma_{xy} = \sqrt{1}$ based on a random regression task with the same architecture.

Panel D illustrates results from running the same task and network configuration but initialised with randomly large weights having a variance of $\sigma = 1$.

In panel E, we trained a two-layer linear network with $N_i = N_h = N_o = 4$ on a random regression task for $\lambda \in [-5,-4,-3,-2,-1,0,1,2,3,4,5]$ to convergence. Subsequently, we added Gaussian noise with $\mu = 0, \sigma \in [0,0.5,1]$ to the inputs (top panel) or synaptic weights (bottom panel) and calculated the expected mean squared error.

A.5.7 Figure 3.5

Fig. 3.5 panel A was generated by training a linear network with $N_i = 5$, $N_h = 10$, $N_o = 5$ on the target $\mathbf{Y}$ as shown in Eq. A.194 (equal diagonal). The network was initialised with $\sigma = 0.1$. The learning rate was $\eta = 0.01$.

Fig. 3.5 panel D, E and F was generated by training a linear network with $N_i = 2$, $N_h = 10$, $N_o = 2$ on the target $\mathbf{Y}$ as shown in Fig. 3.5 C and input $\mathbf{X} = bfi$. The network was initialised with small $\sigma = 0.00001$, intermediate $\sigma = 0.3$ and large $\sigma = 2$ synaptic weights. The learning rate was $\eta = 0.0001$.

A.5.8 Figure 3.6

Panel A illustrates schematic representations of the network architectures considered: from left to right, a funnel network ($N_i = 4$, $N_h = 2$, $N_o = 2$), a square network ($N_i = 4$, $N_h = 4$, $N_o = 4$), and an inverted-funnel network ($N_i = 2$, $N_h = 2$, $N_o = 4$).

Panel B shows the Neural Tangent Kernel (NTK) distance from initialisation, as defined in Fort et al. [91], across the three architectures shown schematically. The kernel distance is calculated as:

$$\mathbf{K}(t) = 1 - \frac{\langle \mathbf{K}_0, \mathbf{K}_t \rangle}{||\mathbf{K}_0||_F ||\mathbf{K}_t||_F}. \tag{A.290}$$

The simulations conducted on the same task with eleven varying initial λ-balanced weights in $[-9,9]$. The regression task parameters were set with $(\sigma = \sqrt{3})$. The task has batch size $N = 10$. The network has with a learning rate of $\eta = 0.01$. The lambda-balanced network are initialised with $\sigma xy = \sqrt{1}$ of a random regression task.

Panel C shows the Neural Tangent Kernel (NTK) distance from initialisation for the funnel architectures shown schematically with dimensions $N_i = 3$, $N_h = 2$, and $N_o = 2$. The simulations conducted on the same task with twenty one varying initial λ-balanced weights in $[-9,9]$. The regression task parameters were set with $(\sigma = \sqrt{3})$. The task has batch size $N = 30$. The network has with a learning rate of $\eta = 0.002$. The lambda-balanced network are initialised with $\sigma xy = \sqrt{1}$ of a random regression task.

A.5.9 Figure 3.7

This figure presents a schematic of the Linear RNN model, which captures task dynamics through time-dependent singular values. The input and output weight matrices are denoted as $\mathbf{W_1}$ and $\mathbf{W_2}$, respectively, with $\mathbf{W}_h$ representing the recurrent hidden weight matrix. The data correlation matrices $\mathbf{\Sigma}^{YX_t}$ maintain constant left and right singular vectors, while their singular values $\mathbf{S_t}$ vary over time.

A.5.10 Figure 3.8

Phase plots depicting the kernel distance of the NTK from initialisation, computed as

$$\mathbf{K}(t) = 1 - \frac{\langle \mathbf{K}_0, \mathbf{K}_t \rangle}{||\mathbf{K}_0||_F ||\mathbf{K}_t||_F} \tag{A.291}$$

as a function of trajectory length, which varies between $[1,7]$, and initialisation scale, which ranges from $[0.01,1]$. The plots compare LRNN models initialised with weights that are (left) unaligned and (right) aligned dicribed bellow. The network architecture consists of input, hidden, and output layers, each containing four units.

A.5.10.1 Structured task dynamics

To create data with input-output correlation matrices that have constant left and right singular vectors and temporally-structured singular value dynamics, we similarly reverse-engineer the equations in Eq. 3.32. We first generate random Gaussian input centered at 0, which is then whitened.

The data singular values $\mathbf{S}_{1:T}$ are created by setting the singular values in each dimension α and at each trajectory timestep t according to the specified task dynamics f (constant $f(\lambda_\alpha,t)=1$, exponential $f(\lambda_\alpha,t)=\lambda_\alpha^t$, inverse-exponential $f(\lambda_\alpha,t)=\lambda_\alpha^{T-t}$) and hyperparameters $\delta_\alpha,\lambda_\alpha$, such that $s_{\alpha,t}=\delta_\alpha f(\lambda_\alpha,t)$. We generate constant left and right singular vectors $\mathbf{U}_2,\mathbf{V}_1$ by taking the SVD of a random matrix. Finally, we create the output according to

$$\boldsymbol{Y}_T=\sum_{i=1}^{T}\mathbf{U}_2\mathbf{S}_t\mathbf{V}_1^\top\boldsymbol{X}_t \tag{A.292}$$

A.5.10.2 Aligned LRNN

To initialise aligned LRNNs, we reverse-engineer the weight matrices starting from the connectivity modes as described in Eq. 3.32. We specify the initialisation of the connectivity modes (input, recurrent, output) as hyperparameters, which are the diagonal matrices $\overline{\mathbf{W}}_1,\overline{\mathbf{W}}_h,\overline{\mathbf{W}}_2$. We then create orthogonal matrices $\mathbf{R}_1$, $\mathbf{R}_2=(\mathbf{R}_1)^{-1}$ and use the left and right singular matrices $\mathbf{U}_2,\mathbf{V}_1$ of the data correlation matrices (see below in Appendix A.5.10.1) to form the weight matrices according to:

$$\mathbf{W}_1 = \mathbf{R}_1\overline{\mathbf{W}}_1\mathbf{V}_1^\top \tag{A.293}$$

$$\mathbf{W}_h = \mathbf{R}_2\overline{\mathbf{W}}_h\mathbf{R}_1^\top \tag{A.294}$$

$$\mathbf{W}_2 = \mathbf{U}_2\overline{\mathbf{W}}_2\mathbf{R}_2^\top \tag{A.295}$$

A.5.10.3 Unaligned LRNN

We create unaligned LRNNs by initialising the weights with a Gaussian distribution of mean 0 and standard deviation $\sigma/\sqrt{N_{\text{in}}}$, where σ is a specified hyperparameter and N_{i} is the row-size of the corresponding weight matrix.

A.5.10.4 Training

Networks are trained using gradient descent on the mean squared error and automated differentiation in order to compare our theoretical results with standard deep learning approaches. We modify our learning timescale in the theory to account for the additional scalars introduced by taking the mean over samples P and output dimension N_o ($\tau = PN_o/\eta$, where η is the learning rate) when comparing to simulation. Training was conducted for 30,000 epochs using a learning rate of 0.1, with 50 samples per run. All layers were trainable, and the kernel distance values were averaged over five different seeds. Throughout training, the singular values remained constant.

A.5.11 Figure 3.9

In panel A, we conducted an experiment to explore the relationship between relative weight scale, absolute weight scale, and the network's learning regime in a general setting. The absolute scale of the weights in Fig. 3.9**A** is defined as the norm of $\mathbf{W}_2\mathbf{W}_1$. Random initial weights with specified relative and absolute scales were generated, and the network was trained on a random input-output task. During training, we calculated the logarithmic kernel distance of the NTK from initialisation and the logarithmic loss. The kernel distance is calculated as: $\mathbf{K}(t) = 1 - \frac{\langle \mathbf{K}_0, \mathbf{K}_t \rangle}{||\mathbf{K}_0||_F ||\mathbf{K}_t||_F}$. as defined in Fort et al. [2020]. These values were visualised as heat maps for

$\lambda \in [-9,9]$ and relative scales in $(0,20]$. The regression task parameters were set with $(\sigma = \sqrt{3})$. The task has batch size $N = 10$. The network has with a learning rate of $\eta = 0.01$. The lambda-balanced network are initialised with $\mathbf{E}[xy^T] = I$ of a random regression task. Fig. 3.9**A** shows that a square ($N_i = N_c$) linear neural network satisfying Assumption 2 (λ-balanced) only show amplifying dynamics when the weights are balanced with small $|\lambda|$.

In panel B, the setup is identical to that of Fig. 3.9**A** except a few changes. We used the linear neural network with $N_i = 20, N_h = 20, N_o = 2$. We randomly initialised $\mathbf{W}_2, \mathbf{W}_1$ (which do not satisfying the balanced condition) using symmetrised initialisation [51] to set the initial function equal to the zero function. The layer imbalance was implemented using the standard deviations σ_1, σ_2 used to initialise the weights. To change the weight-to-target ratio, we used target downscaling in the range $[0.1, 50]$. The inputs are not whitened, as the dynamics remain largely independent of the input distribution. Being so close to the target results in negligible learning dynamics.

Code The codes used to plot Fig. 3.9 are based on the original codes of Domine et al. [2024] with only minor changes. They are available at `https://anonymous.4open.science/r/linear_first-0FCA`.

A.5.12 Figure 3.10

Panels C-F in Fig. 3.10 were generated by training a linear network with $N_i = 8$, $N_h = 14$, $N_o = 8$ on the $N = 8$ items of the semantic hierarchy task. The learning rate was $\eta = 0.05$ and the initial weights in panels C, D, and E were sampled from a normal distribution with $\sigma = 0.0001$ and $\sigma = 0.42$ and zero-balanced weights with $\sigma = 0.44$ respectively.

A.5.13 Figure 3.11

Fig. 3.11 panel A was generated by training a linear network with $N_i = 5$, $N_h = 10$, $N_o = 6$ subsequently on four different random regression tasks with $N = 25$. The learning rate was $\eta = 0.05$ and the initial weights were small ($\sigma = 0.0001$).

Panels B were generated by running 50 simulations on two subsequent random regression tasks, each with a different initial random seed. The simulation was repeated three times, the first time with a linear, the second time with a tanh and the last time with a ReLU activation function in the hidden layer. Dimension were randomly sampled such that $N_i \in [2,30]$, $N_o \in [2,30]$, $N_h = [\min(N_i,N_o),30]$ and $N = 100$. The standard deviation of the initial weight was chosen such that $\sigma = \frac{0.5}{\sqrt{0.5(N_i+N_h)}}$. The learning rate was $\eta = 0.075$.

A.5.14 Figure 3.12

Fig. 3.12 panel A was generated by training a linear network with $N_i = 4$, $N_h = 6$, $N_o = 4$ on a reversal learning task (see Sec. 3.5.2), which was derived from a random regression task. The learning rate was $\eta = 0.05$ and initial weights had a standard deviation of $\sigma = 0.25$. Panel B was generated by training a shallow linear network (see Sec. A.4.2.1) on the same reversal learning task, with identical hyperparameters as in panel A.

For the top and bottom rows of panels E-F a linear network with $N_i = 8$, $N_h = 14$, $N_o = 8$ was trained on the semantic hierarchy task, followed by training the network on the adapted semantic hierarchy as depicted in Fig. 3.12 C top, which is a reversal learning task and the colour hierarchy respectively. The learning rate was $\eta = 0.05$ and σ was set to 0.001 and 0.35 respectively.

Appendix B

Appendix Chapter 4

B.1 Additional entropy phase diagrams

In Fig. 4.4 we show phase diagrams of the aggregate entropy as a function of initialisation parameters, for both ReLU and sigmoidal networks. In Fig. B.1 below, we show additional plots with the individual entropy terms (H_u defined over the unit activations, and H_h defined over the head weights).

B.2 Diversity of forgetting curves

B.3 Implementation and simulation

B.3.1 Figure 4.1

Panel A illustrates the teacher-student setup, where a student network is trained using labels generated by a fixed teacher network.

Panel B shows the student-teacher continual learning set up. To model continual learning, a two-layer student network is trained sequentially on two distinct teacher networks, each corresponding to a different task (Task 1 and Task 2). In this setup, the student learns from the outputs of the teacher networks. The two student instances share the same input weights but have separate output heads. The student's initial weights, denoted by I_W and I_h, are parameterised by Θ_W and Θ_h, respectively. This figure is adapted from Fig. 1 in Lee et al. [167].

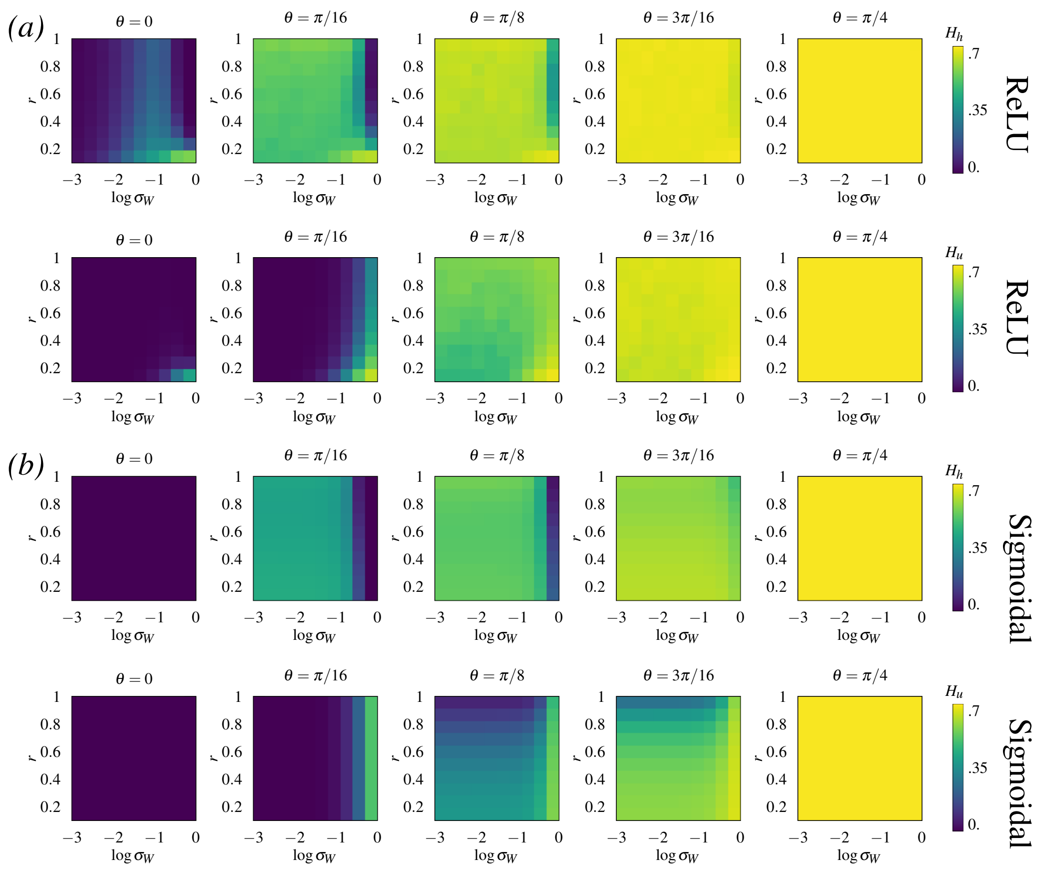


Figure B.1: Additional Phase Diagrams. Here we show the equivalent phase diagrams from Fig 4.4 for entropy measures over the unit activations and head weights.

B.3.2 Figure 4.2

Schematic representation of specialisation in the student-teacher setup. Saad and Solla [238] showed that, when both teacher and student are modelled as committee machines, each student neuron (yellow and blue) specialises by aligning with a specific teacher neuron (yellow and blue). Similarly, Goldt et al. [106] observed that for certain activation functions in two-layer networks, an over-parameterised student will selectively use only a subset of those units to replicate the teacher's outputs. This phenomenon, termed specialisation, stands in contrast to a student redundantly sharing representations of the teacher across neurons (green).

B.3.3 Figure 4.3

Data Preparation

The dataset is sampled from an independent and identically distributed (i.i.d.) Gaussian distribution with a mean of 0 and unit variance, assuming an infinite dataset size. Both the teacher and student networks have the same input dimension and a single output dimension.

Model Architecture

The teacher network consists of two hidden units with no bias, and its weights are initialised from a Gaussian distribution. The student network has four hidden units, also without bias. Its initialisation differs based on activation function: the second-layer weights have low variance and magnitude for sigmoidal activation, while they have high variance for ReLU activation.

Training Process

Training is performed using stochastic gradient descent (SGD) with a batch size of 1 for a total of 80 million steps. The optimisation process is guided by the mean squared error (MSE) loss function. Both the hidden and head layers of the student network are trainable, ensuring full network adaptation during training. The head layer remains unchanged throughout training, as it is neither reset nor copied.

B.3.4 Figure 4.4

Data Preparation

The phase diagrams depict the aggregated entropy for various initialisations, with color indicating different entropy levels. The dataset is drawn from an independent and identically distributed (i.i.d.) Gaussian distribution with a mean of 0 and unit variance, assuming an infinite dataset size.

Model Architecture

Both the teacher and student networks share the same input dimension of 10,000 and an output dimension of 1. The networks utilise ReLU and sigmoidal nonlinearities. The teacher has 1 hidden units, no bias, and initial weights sampled from a Gaussian distribution with a mean of 0 and a standard deviation of 0.001. The student has two hidden units, no bias. The student is intialised with the polar cordinate methode discibed in Eq. 4.24. The x-axis represents the standard deviation of the first layer's initialisation, while the y-axis corresponds to the norm of the second-layer initialisation, expressed in polar coordinates. Different panels show varying angles, ranging from orthogonal units ($\theta = 0$) to identical units ($\theta = \pi/4$). Blue-leaning initialisations indicate specialised solutions, whereas yellow-leaning ones correspond to high-entropy, non-specialised configurations.

Training Process

The training process is conducted using stochastic gradient descent (SGD) as the optimiser, with a learning rate of 1.0. The model is trained for a total of 1,000,000 steps using a batch size of 1. The mean squared error (MSE) loss function is employed to guide the optimisation process. Both the hidden and head layers of the student network are trainable, ensuring that the entire network undergoes adaptation during training. Additionally, the head layer is not reset or copied at any stage of training.

B.3.5 Figure 4.5

Data Preparation

The dataset is generated from independent and identically distributed (i.i.d.) sample of a Gaussian distribution with a mean of 0 and variance of 1, assuming an infinite dataset size.

Model Architecture

Both the teacher and student networks have an input dimension of 10,000 and an output dimension of 1. The networks utilise sigmoidal nonlinearities. The teacher network consists of a single hidden unit, no bias and initial weights sampled from a Gaussian distribution with a mean of 0. The student network has two hidden units and no bias. The initialisation follows the polar coordinate method outlined in Eq. 4.24, as shown in the accompanying figure. The weights of the first layer are initialised with zero mean and a variance of 0.001. The similarity between the teacher is controled by Eq. 4.2.

Training Process

Training is performed using stochastic gradient descent (SGD) with a learning rate of 1.0, and the model is trained until convergence. The mean squared error (MSE) loss function is used to guide the learning process. Both the hidden and head layers of the student network are trainable, allowing the entire network to adapt throughout training.

B.3.6 Figure 4.6

Data Preparation

The input data is sampled from an independent and identically distributed (i.i.d.) Gaussian distribution with zero mean and unit variance, assuming an infinite dataset size. No noise is added to the student inputs or teacher outputs. The input dimension is set to 10,000 and the output dimension is 1 for both student and teacher networks. Two teachers are used in this continual learning setup.

Model Architecture

The teacher network has a single hidden unit and no bias, with weights initialised from a Gaussian distribution with zero mean and unit variance. The student network consists of two hidden units, also without bias. The input weights are initialised from a Gaussian with mean 0 and standard deviation 0.001. The head of the student network is initialised using the polar method, with different norms and angles assigned for each task, ensuring task-specific readouts. The input layer is shared across tasks, while the output heads are task-specific.

Training Process

Training is performed using stochastic gradient descent (SGD) with a batch size of 1 and a fixed learning rate of 1.0, for a total of 10 million steps. The mean squared error (MSE) is used as the loss function. Both the hidden and head layers of the student network are trainable. At the task switch point, the head is not copied. EWC is applied for continual learning, with node-level importance and a regularisation strength shown in the figure.

B.3.7 Figure 4.7

We conduct our experiments using open-source frameworks [177, 1]. Specifically, we implement a beta-VAE with the ”DeepGaussianLinear” architecture for the decoder and ”DeepLinear” for the encoder. We modify the Xavier initialisation where the weights of the linear layers will have values sampled from $U(-a, a)$ with

$$a = \text{gain} \times \sqrt{\frac{6}{\text{fan\_in} + \text{fan\_out}}}$$

We vary the gain between 0.3 and 3 and run each experiment over 4 seeds. All network parameters are set to their default values as provided by the respective open-source frameworks. We run the experiments for 20 Epochs and 157499 iterations.

DCI Disentanglement

Eastwood and Williams [80] define three key properties of learned representations: Disentanglement, Completeness, and Informativeness. To assess these, they calculate the importance of each dimension of the representation in predicting a factor of variation. This can be done using models like Lasso or Random Forest classifiers. Disentanglement is computed by subtracting the entropy of the probability that a representation dimension predicts a factor, weighted by its relative importance. Completeness is similarly measured, focusing on how well a factor is captured by the dimensions. Informativeness is evaluated as the prediction error of the factors. We use the implementation in Locatello et al. [177]. In this implementation, we sample 10,000 training and 5,000 test points, then use gradient-boosted trees from Scikit-learn to obtain feature importance weights. These weights form an importance matrix, with rows representing factors and columns representing dimensions. Disentanglement is calculated by normalising the columns of this matrix, subtracting the entropy from 1 for each column, and then weighting by each dimension's relative importance.

B.3.8 Figure 4.8

In our experiments the VAE has 16 hidden neurons. These 16 neurons become the input (and output) to the SAE. The SAE then projects this up to a latent space of dimension 2048 which has a ReLU activation function. For our baseline, we train the SAE with the typical L2 reconstruction loss and *L1 regularisation on the hidden activity*. We calculate the sparsity across the dataset as the average number of datapoints the hidden neurons respond to, over the 60000 datapoints:

$$\frac{1}{2048}\sum_{i=1}^{2048}\sum_{j=1}^{60000}\mathbf{1}(H_{ij}) \tag{B.1}$$

To initialise the layers of the iSAE and SAE we define an imbalance parameter υ (note that this is not the same hyper-parameter as the λ notation employed in the main text and is defined purely for practicality in this experiment). The encoder weights are initialised by sampling from a Gaussian with standard deviation $\sigma = 0.001\frac{1}{\upsilon}$. The decoder weights are sampled from a Gaussian with standard deviation $\sigma = 0.001\upsilon$. Thus, as υ increases the decoder is initialised with increasingly large weights compared to the decoder.

B.3.9 Figure 4.9

Gabor Filters

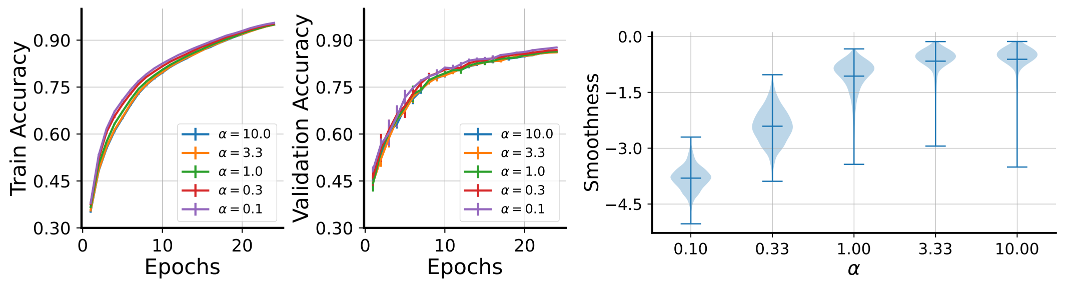
We are training a small ResNet based on the CIFAR10 script provided in the DAWN benchmark (code available here). The only modifications to the provided code base are we increase the convolution kernel size from 3×3 to 15×15, to better observe the learned spatial patterns, and we set the weight decay parameter to 0 to avoid confounding variables. Moreover, we are dividing the convolutional filters weights by a parameter α (after standard initialisation) which controls the balancedness of the network. To quantify the smoothness of the filters, we compute the normalised Laplacian of each filter $w_{ij} \in \mathbb{R}^{15\times 15}$, over input $i = (1,2,3)$ and output $j = (1,...,64)$

channels

$$\text{smoothness}(w_{ij}) := \left\| \frac{w_{ij}}{\|w_{ij}\|_2} * \Delta \right\|_2^2 \tag{B.2}$$

where the Laplacian kernel is defined as

$$\Delta := \begin{pmatrix} -0.25 & -0.5 & -0.25 \\ -0.5 & 2 & -0.5 \\ -0.25 & -0.5 & -0.25 \end{pmatrix}. \tag{B.3}$$



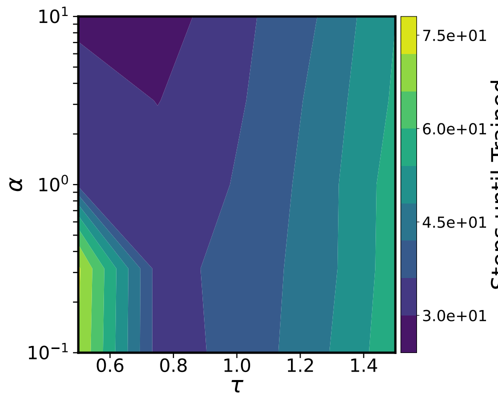
(a) Accuracy **(b)** Gabor Smoothness

Figure B.3: Interpreting convolutional filters. CNN experiments on CIFAR10. We can see in **A)** that all networks achieve comparable training and test accuracy, despite the modification in initialisation. However, in **B)** we see that networks with a small initialisation ($\alpha < 1$) learn much smoother filters, giving quantiative support to results in Fig. 4.9. The smoothness is defined as the normalised Laplacian of the filters (see text, Eq. B.2).

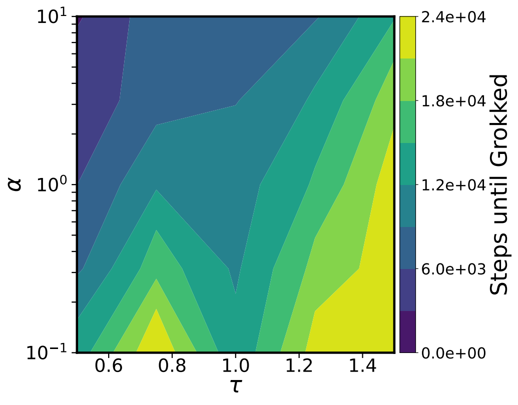
Grokking

We are training a one layer transformer model on the modular arithmetic task in Power et al. [218]. Our experimental code is based on an existing Pytorch implementation (code available here). The only modifications to the provided code base is that we use a single transformer layer (instead of the default 2-layer model). Prior analysis in Nanda et al. [204] has shown that this model can learn a minimal (attention-based) circuit that solves the task.

We study the effects on grokking time (defined as ≥ 0.99 accuracy on the validation data) of two manipulations. Firstly, we divide the embedding weights of the positional and token embeddings by the same balancedness parameter α as in the CNN gabor experiments. Secondly, like in Kumar et al. [155], we multiply the output of the model (i.e., the logits) by a factor τ and divide the learning rate by τ^2.



(a) Steps until training



(b) Steps until grokked

Figure B.4: Transformer Grokking in Modular Arithmetic Task. A) Shows the number of training steps required until the training accuracy passes a predefined threshold of 99%; we sample scaling $\tau \in \{0.5, 0.75, 1.0, 1.25, 1.5\}$ [155] and balance $\alpha \in \{0.1, 0.3, 1.0, 3.0, 10\}$ on a regular grid over $n = 5$ random initialisations with a maximal computational budget of $m = 30,000$ training steps. **B)** Same as **A)**, but reporting the number of training steps required until the test performance passes the predefined threshold of 99%. We clearly see the fastest grokking in an unbalanced rich setting.

B.3.10 Figure 4.10

This experiment involves training a neural network on a subset of the MNIST dataset and analysing how different nodes contribute to learning. The key steps in the process are as follows:

Data Preparation

The MNIST dataset, which consists of handwritten digits, is downloaded and transformed into tensors. A subset of the dataset is created by selecting only specific digit classes (e.g., digits 0-4 for initial training). The selected data is split into training (80%) and validation (20%) sets, and corresponding data loaders are created.

Model Architecture

A two layer neural network is defined with an input layer of 784 neurons (corresponding to 28×28 pixel images), one hidden layer, and two output layers. The activation function used is the sigmoid function. The second-layer weights are initialised using a combination of uniform and normal distributions, with different variance levels.

Training Process

The model is trained using stochastic gradient descent (SGD) with a momentum factor. The experiment is run multiple times with different random seeds using multiprocessing, ensuring robustness in results. The initial task involves training on the base dataset (digits 0-4).

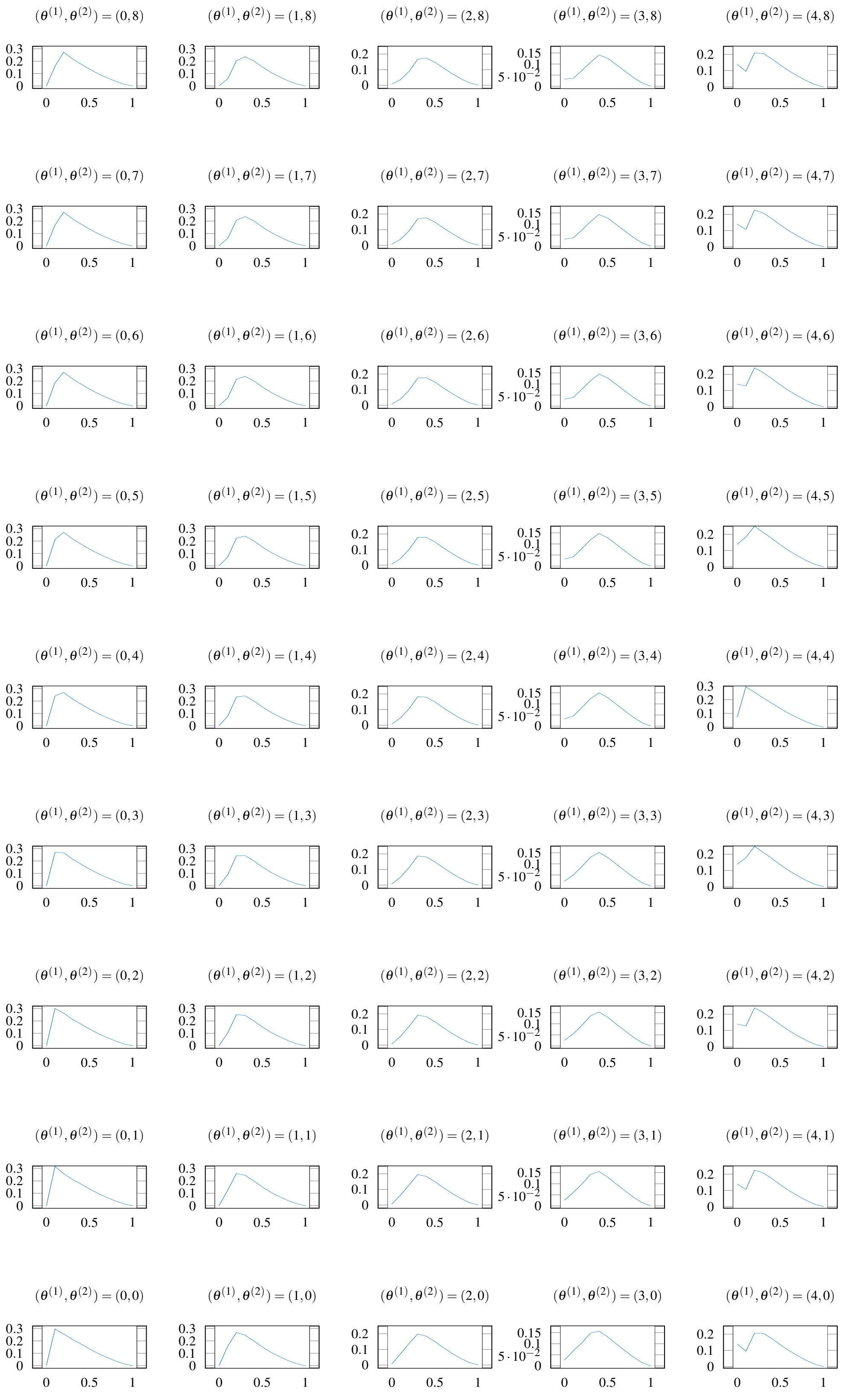


Figure B.2: Initialisation can lead to a diversity of specialisation dynamics and a diversity of relationships between forgetting and task similarity. R, σ_W fixed, $\theta^{(1)}, \theta^{(2)}$ measured in increments of $\pi/16$. Scaled error function, $P^* = 1, P = 1$.

Bibliography

[1] Amir H. Abdi, Purang Abolmaesumi, and Sidney Fels. Variational learning with disentanglement-pytorch. *arXiv preprint arXiv:1912.05184*, 2019.

[2] Daron Acemoglu. Harms of ai. Technical report, National Bureau of Economic Research, 2021.

[3] Urte Adomaityte, Gabriele Sicuro, and Pierpaolo Vivo. Classification of superstatistical features in high dimensions. In *2023 Conference on Neural Information Procecessing Systems*, 2023.

[4] Madhu S Advani, Andrew M Saxe, and Haim Sompolinsky. High-dimensional dynamics of generalization error in neural networks. *Neural Networks*, 132: 428–446, 2020.

[5] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.

[6] Kwangjun Ahn, Xiang Cheng, Minhak Song, Chulhee Yun, Ali Jadbabaie, and Suvrit Sra. Linear attention is (maybe) all you need (to understand transformer optimization). *arXiv preprint arXiv:2310.01082*, 2023.

[7] Andrey Alekseev and Anatoly Bobe. Gabornet: Gabor filters with learnable parameters in deep convolutional neural network. In *2019 International Conference on Engineering and Telecommunication (EnT)*, pages 1–4. IEEE, 2019.

[8] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. *Advances in neural information processing systems*, 32, 2019.

[9] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *International conference on machine learning*, pages 242–252. PMLR, 2019.

[10] D. Amodei, D. Hernandez, G. Sastry, C. Jack, G. Brockman, and I. Sutskever. Ai and compute, 2018. URL `https://openai.com/index/ai-and-compute/`. Retrieved 2024-08-15.

[11] Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, et al. Massively multilingual neural machine translation in the wild: Findings and challenges. *arXiv preprint arXiv:1907.05019*, 2019.

[12] Sanjeev Arora, Nadav Cohen, Noah Golowich, and Wei Hu. A convergence analysis of gradient descent for deep linear neural networks. *arXiv preprint arXiv:1810.02281*, 2018.

[13] Sanjeev Arora, Nadav Cohen, and Elad Hazan. On the optimization of deep networks: Implicit acceleration by overparameterization. In *International Conference on Machine Learning*, pages 244–253. PMLR, 2018.

[14] Sanjeev Arora, Nadav Cohen, Wei Hu, and Yuping Luo. Implicit regularization in deep matrix factorization. *Advances in Neural Information Processing Systems*, 32, 2019.

[15] Sanjeev Arora, Simon Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning*, pages 322–332. PMLR, 2019.

[16] Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, Russ R Salakhutdinov, and Ruosong Wang. On exact computation with an infinitely wide neural net. *Advances in Neural Information Processing Systems*, 32, 2019.

[17] Haruka Asanuma, Shiro Takagi, Yoshihiro Nagano, Yuki Yoshida, Yasuhiko Igarashi, and Masato Okada. Statistical mechanical analysis of catastrophic forgetting in continual learning with teacher and student networks. *Journal of the Physical Society of Japan*, 90(10):104001, 2021.

[18] Alexander Atanasov, Blake Bordelon, and Cengiz Pehlevan. Neural networks as kernel learners: The silent alignment effect. In *International Conference on Learning Representations*, 2022.

[19] Benjamin Aubin, Antoine Maillard, Florent Krzakala, Nicolas Macris, Lenka Zdeborová, et al. The committee machine: Computational to statistical gaps in learning a two-layers neural network. *Advances in Neural Information Processing Systems*, 31, 2018.

[20] Bruno B Averbeck, Peter E Latham, and Alexandre Pouget. Neural correlations, population coding and computation. *Nature reviews neuroscience*, 7(5): 358–366, 2006.

[21] Shahar Azulay, Edward Moroshko, Mor Shpigel Nacson, Blake E Woodworth, Nathan Srebro, Amir Globerson, and Daniel Soudry. On the implicit bias of initialization shape: Beyond infinitesimal mirror descent. In *International Conference on Machine Learning*, pages 468–477. PMLR, 2021.

[22] Yasaman Bahri, Jonathan Kadmon, Jeffrey Pennington, Sam S Schoenholz, Jascha Sohl-Dickstein, and Surya Ganguli. Statistical mechanics of deep learning. *Annual Review of Condensed Matter Physics*, 11:501–528, 2020.

[23] Pierre Baldi and Kurt Hornik. Neural networks and principal component analysis: Learning from examples without local minima. *Neural networks*, 2 (1):53–58, 1989.

[24] Andrea Banino, Caswell Barry, Benigno Uria, Charles Blundell, Timothy Lillicrap, Piotr Mirowski, Alexander Pritzel, Martin J Chadwick, Thomas Degris, Joseph Modayil, et al. Vector-based navigation using grid-like representations in artificial agents. *Nature*, 557(7705):429–433, 2018.

[25] Omri Barak. Recurrent neural networks as versatile tools of neuroscience research. *Current opinion in neurobiology*, 46:1–6, 2017.

[26] Horace B Barlow. Single units and sensation: a neuron doctrine for perceptual psychology? *Perception*, 1(4):371–394, 1972.

[27] Caswell Barry, Lin Lin Ginzberg, John O'Keefe, and Neil Burgess. Grid cell firing patterns signal environmental novelty by expansion. *Proceedings of the National Academy of Sciences*, 109(43):17687–17692, 2012.

[28] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. Reconciling modern machine-learning practice and the classical bias–variance trade-off. *Proceedings of the National Academy of Sciences*, 116(32):15849–15854, 2019.

[29] Gerard Ben Arous, Reza Gheissari, and Aukosh Jagannath. High-dimensional limit theorems for sgd: Effective dynamics and critical scaling. *Advances in Neural Information Processing Systems*, 35:25349–25362, 2022.

[30] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48, 2009.

[31] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[32] Marcus K Benna and Stefano Fusi. Place cells may simply be memory cells: Memory compression leads to spatial tuning and history dependence.

Proceedings of the National Academy of Sciences, 118(51):e2018422118, 2021.

[33] Silvia Bernardi, Marcus K Benna, Mattia Rigotti, Jérôme Munuera, Stefano Fusi, and C Daniel Salzman. The geometry of abstraction in the hippocampus and prefrontal cortex. *Cell*, 183(4):954–967, 2020.

[34] Michael Biehl and Holm Schwarze. Learning by on-line gradient descent. *Journal of Physics A: Mathematical and general*, 28(3):643, 1995.

[35] Colin Blakemore, James PJ Muncey, and Rosalind M Ridley. Stimulus specificity in the human visual system. *Vision research*, 13(10):1915–1931, 1973.

[36] Blake Bordelon and Cengiz Pehlevan. Self-consistent dynamical field theory of kernel evolution in wide neural networks. *Advances in Neural Information Processing Systems*, 35:32240–32256, 2022.

[37] Blake Bordelon and Cengiz Pehlevan. Deep linear network training dynamics from random initialization: Data, width, depth, and hyperparameter transfer, 2025. URL `https://arxiv.org/abs/2502.02531`.

[38] Lara M Boyle, Lorenzo Posani, Sarah Irfan, Steven A Siegelbaum, and Stefano Fusi. Tuned geometries of hippocampal representations meet the computational demands of social memory. *Neuron*, 112(8):1358–1371, 2024.

[39] Lukas Braun, Clémentine Carla Juliette Dominé, James E Fitzgerald, and Andrew M Saxe. Exact learning dynamics of deep linear networks with prior knowledge. In *Advances in Neural Information Processing Systems*, 2022.

[40] Lukas Braun, Christopher Summerfield, and Andrew Saxe. Preserving knowledge during learning [unpublished manuscript]. *Department of Experimental Psychology, University of Oxford*, 2024.

[41] Chris Burgess and Hyunjik Kim. 3d shapes dataset. https://github.com/deepmind/3dshapes-dataset/, 2018.

[42] Christopher P Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in β-vae. *arXiv preprint arXiv:1804.03599*, 2018.

[43] Neil Burgess, Caswell Barry, and John O'keefe. An oscillatory interference model of grid cell firing. *Hippocampus*, 17(9):801–812, 2007.

[44] S.E. Carey. *Conceptual Change In Childhood*. MIT Press, Cambridge, MA, 1985.

[45] Rich Caruana. Multitask learning. *Machine learning*, 28:41–75, 1997.

[46] Romain D Cazé. All neurons can perform linearly non-separable computations. *F1000Research*, 10:539, 2022.

[47] Rishidev Chaudhuri, Berk Gerçek, Biraj Pandey, Adrien Peyrache, and Ila Fiete. The intrinsic attractor manifold and population dynamics of a canonical cognitive circuit across waking and sleep. *Nature neuroscience*, 22(9):1512–1520, 2019.

[48] Guifen Chen, John Andrew King, Yi Lu, Francesca Cacucci, and Neil Burgess. Spatial cell firing during virtual navigation of open arenas by head-restrained mice. *Elife*, 7, 2018.

[49] Ricky T. Q. Chen, Xuechen Li, Roger Grosse, and David Duvenaud. Isolating sources of disentanglement in variational autoencoders, 2019. URL `https://arxiv.org/abs/1802.04942`.

[50] Lenaic Chizat and Francis Bach. Implicit bias of gradient descent for wide two-layer neural networks trained with the logistic loss. In *Conference on learning theory*, pages 1305–1338. PMLR, 2020.

[51] Lenaic Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. *Advances in neural information processing systems*, 32, 2019.

[52] SueYeon Chung and Larry F Abbott. Neural population geometry: An approach for understanding biological and artificial neural networks. *Current opinion in neurobiology*, 70:137–144, 2021.

[53] Uri Cohen, SueYeon Chung, Daniel D Lee, and Haim Sompolinsky. Separability and geometry of object manifolds in deep neural networks. *Nature communications*, 11(1):746, 2020.

[54] Edo Cohen-Karlik, Itamar Menuhin-Gruman, Raja Giryes, Nadav Cohen, and Amir Globerson. Learning low dimensional state spaces with overparameterized recurrent neural nets. In *The Eleventh International Conference on Learning Representations*, 2023. URL `https://openreview.net/forum?id=k9CF4h3muD`.

[55] Hristos S Courellis, Juri Minxha, Araceli R Cardenas, Daniel L Kimmel, Chrystal M Reed, Taufik A Valiante, C Daniel Salzman, Adam N Mamelak, Stefano Fusi, and Ueli Rutishauser. Abstract representations emerge in human hippocampal neurons during inference. *Nature*, pages 1–9, 2024.

[56] Francis Crick. The recent excitement about neural networks. *Nature*, 337 (6203):129–132, 1989.

[57] Christopher J Cueva and Xue-Xin Wei. Emergence of grid-like representations by training recurrent neural networks to perform spatial localization. *arXiv preprint arXiv:1803.07770*, 2018.

[58] Hugo Chao Cui. Topics in statistical physics of high-dimensional machine learning. Technical report, EPFL, 2024.

[59] Sander Dalm, Joshua Offergeld, Nasir Ahmad, and Marcel van Gerven. Efficient deep learning with decorrelated backpropagation. *arXiv preprint arXiv:2405.02385*, 2024.

[60] Matthias De Lange, Rahaf Aljundi, Marc Masana, Sarah Parisot, Xu Jia, Aleš Leonardis, Gregory Slabaugh, and Tinne Tuytelaars. A continual learning

survey: Defying forgetting in classification tasks. *IEEE transactions on pattern analysis and machine intelligence*, 44(7):3366–3385, 2021.

[61] Li Deng. The mnist database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine*, 29(6):141–142, 2012.

[62] Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. Qlora: Efficient finetuning of quantized llms. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 10088–10115. Curran Associates, Inc., 2023. URL `https://proceedings.neurips.cc/paper_files/paper/2023/file/1feb87871436031bdc0f2beaa62a049b-Paper-Conference.pdf`.

[63] Oussama Dhifallah and Yue M Lu. Phase transitions in transfer learning for high-dimensional perceptrons. *Entropy*, 23(4):400, 2021.

[64] Thang Doan, Mehdi Abbana Bennani, Bogdan Mazoure, Guillaume Rabusseau, and Pierre Alquier. A theoretical analysis of catastrophic forgetting through the ntk overlap matrix. In *International Conference on Artificial Intelligence and Statistics*, pages 1072–1080. PMLR, 2021.

[65] Christian F Doeller, Caswell Barry, and Neil Burgess. Evidence for grid cells in a human memory network. *Nature*, 463(7281):657–661, 2010.

[66] Adrien Doerig, Rowan P Sommers, Katja Seeliger, Blake Richards, Jenann Ismael, Grace W Lindsay, Konrad P Kording, Talia Konkle, Marcel AJ Van Gerven, Nikolaus Kriegeskorte, et al. The neuroconnectionist research programme. *Nature Reviews Neuroscience*, 24(7):431–450, 2023.

[67] Clémentine CJ Dominé, Nicolas Anguita, Alexandra M Proca, Lukas Braun, Daniel Kunin, Pedro AM Mediano, and Andrew M Saxe. From lazy to rich: Exact learning dynamics in deep linear networks. *arXiv preprint arXiv:2409.14623*, 2024.

[68] Clementine CJ Domine, Rodrigo Antonio Carrasco Davis, Luke Hollingsworth, Nikoloz Sirmpilatze, Adam L Tyson, Devon Jarvis, Caswell Barry, and Andrew M Saxe. Neuralplayground: A standardised environment for evaluating models of hippocampus and entorhinal cortex. *bioRxiv*, pages 2024–03, 2024.

[69] Clémentine C.J. Dominé, Lukas Braun, James E. Fitzgerald, and Andrew M. Saxe. Exact learning dynamics of deep linear networks with prior knowledge. *Journal of Statistical Mechanics: Theory and Experiment*, 2023(11):114004, 2023.

[70] Kenji Doya. Universality of fully-connected recurrent neural networks. *IEEE Transactions on Neural Networks*, 1993.

[71] Laura N Driscoll, Krishna Shenoy, and David Sussillo. Flexible multitask computation in recurrent networks utilizes shared dynamical motifs. *Nature Neuroscience*, 27(7):1349–1363, 2024.

[72] Simon Du and Wei Hu. Width provably matters in optimization for deep linear neural networks. In *International Conference on Machine Learning*, pages 1655–1664. PMLR, 2019.

[73] Simon Du, Jason Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. In *International conference on machine learning*, pages 1675–1685. PMLR, 2019.

[74] Simon S Du, Wei Hu, and Jason D Lee. Algorithmic regularization in learning deep homogeneous models: Layers are automatically balanced. *Advances in Neural Information Processing Systems*, 31, 2018.

[75] Simon S Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. *arXiv preprint arXiv:1810.02054*, 2018.

[76] Xiaobiao Du, Haiyang Sun, Shuyun Wang, Zhuojie Wu, Hongwei Sheng, Jiaying Ying, Ming Lu, Tianqing Zhu, Kun Zhan, and Xin Yu. 3drealcar: An in-the-wild rgb-d car dataset with 360-degree views, 2024. URL `https://arxiv.org/abs/2406.04875`.

[77] Shiv Ram Dubey, Satish Kumar Singh, and Bidyut Baran Chaudhuri. Activation functions in deep learning: A comprehensive survey and benchmark. *Neurocomputing*, 503:92–108, 2022.

[78] Alexis M. Dubreuil, Adrian Valente, Manuel Beirán, Francesca Mastrogiuseppe, and Srdjan Ostojic. The role of population structure in computations through neural dynamics. *Nature Neuroscience*, 25:783 – 794, 2022.

[79] Jacob Dunefsky, Philippe Chlenski, and Neel Nanda. Transcoders find interpretable llm feature circuits. *URL https://arxiv. org/abs/2406.11944*, 2024.

[80] Cian Eastwood and Christopher KI Williams. A framework for the quantitative evaluation of disentangled representations. In *6th International Conference on Learning Representations*, 2018.

[81] Melikasadat Emami, Mojtaba Sahraee-Ardakan, Parthe Pandit, Sundeep Rangan, and Alyson K Fletcher. Implicit bias of linear rnns. In *International Conference on Machine Learning*, pages 2982–2992. PMLR, 2021.

[82] Andreas Engel. *Statistical mechanics of learning*. Cambridge University Press, 2001.

[83] Tatiana A. Engel, Warasinee Chaisangmongkon, David J. Freedman, and Xiao-Jing Wang. Choice-correlated activity fluctuations underlie learning of neuronal category representation. *Nature Communications*, 6, 2015.

[84] Burak Erdeniz and Nart Bedin Atalay. Simulating probability learning and probabilistic reversal learning using the attention-gated reinforcement learning

(agrel) model. In *The 2010 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6. IEEE, 2010.

[85] Dumitru Erhan, Yoshua Bengio, Aaron Courville, and Pascal Vincent. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341 (3):1, 2009.

[86] Dumitru Erhan, Aaron Courville, Yoshua Bengio, and Pascal Vincent. Why does unsupervised pre-training help deep learning? In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 201–208. JMLR Workshop and Conference Proceedings, 2010.

[87] Matthew Farrell, Stefano Recanatesi, and Eric Shea-Brown. From lazy to rich to exclusive task representations in neural networks and neural codes. *Current Opinion in Neurobiology*, 83:102780, 2023. doi: 10.1016/j.conb.2023.102780.

[88] Matthew Todd Farrell, Stefano Recanatesi, Timothy Moore, Guillaume Lajoie, and Eric Shea-Brown. Gradient-based learning drives robust representations in recurrent neural networks by balancing compression and expansion. *Nature Machine Intelligence*, 4:564 – 573, 2022.

[89] Timo Flesch, Jan Balaguer, Ronald Dekker, Hamed Nili, and Christopher Summerfield. Comparing continual task learning in minds and machines. *Proceedings of the National Academy of Sciences*, 115(44):E10313–E10322, 2018. doi: 10.1073/pnas.1800755115. URL `https://www.pnas.org/content/115/44/E10313`.

[90] Timo Flesch, Keno Juechems, Tsvetomira Dumbalska, Andrew Saxe, and Christopher Summerfield. Orthogonal representations for robust context-dependent task performance in brains and neural networks. *Neuron*, 2022.

[91] Stanislav Fort, Gintare Karolina Dziugaite, Mansheej Paul, Sepideh Kharaghani, Daniel M Roy, and Surya Ganguli. Deep learning versus kernel

learning: an empirical study of loss landscape geometry and the time evolution of the neural tangent kernel. *Advances in Neural Information Processing Systems*, 33:5850–5861, 2020.

[92] Robert M French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3(4):128–135, 1999.

[93] Kenji Fukumizu. Effect of batch learning in multilayer neural networks. *IEEE Transactions on Neural Networks*, 11:17–26, 1998. doi: 10.1109/72.822506.

[94] Marco Fumero, Luca Cosmo, Simone Melzi, and Emanuele Rodola. Learning disentangled representations via product manifold projection. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 3530–3540. PMLR, 18–24 Jul 2021. URL `https://proceedings.mlr.press/v139/fumero21a.html`.

[95] Marco Fumero, Emanuele Rodolá, Clementine Domine, Francesco Locatello, Karolina Dziugaite, and Caron Mathilde. Preface of unireps: the first workshop on unifying representations in neural models. In *Proceedings of UniReps: the First Workshop on Unifying Representations in Neural Models*, pages 1–10. PMLR, 2024.

[96] Stefano Fusi, Earl K Miller, and Mattia Rigotti. Why neurons mix: high dimensionality for higher cognition. *Current opinion in neurobiology*, 37: 66–74, 2016.

[97] Elizabeth Gardner and Bernard Derrida. Three unfinished works on the optimal storage capacity of networks. *Journal of Physics A: Mathematical and General*, 22(12):1983, 1989.

[98] Jesse P Geerts, Fabian Chersi, Kimberly L Stachenfeld, and Neil Burgess. A general model of hippocampal and dorsal striatal learning and decision making. *Proceedings of the National Academy of Sciences*, 117(49):31427–31437, 2020.

[99] Mario Geiger, Stefano Spigler, Arthur Jacot, and Matthieu Wyart. Disentangling feature and lazy training in deep neural networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2020(11):113301, 2020.

[100] Apostolos P Georgopoulos, Andrew B Schwartz, and Ronald E Kettner. Neuronal population coding of movement direction. *Science*, 233(4771):1416–1419, 1986.

[101] Federica Gerace, Luca Saglietti, Stefano Sarao Mannelli, Andrew Saxe, and Lenka Zdeborová. Probing transfer learning with a model of synthetic correlated datasets. *Machine Learning: Science and Technology*, 2022.

[102] Federica Gerace, Diego Doimo, Stefano Sarao Mannelli, Luca Saglietti, and Alessandro Laio. How to choose the right transfer learning protocol? a qualitative analysis in a controlled set-up. *Transactions on Machine Learning Research*, 2024.

[103] Gauthier Gidel, Francis Bach, and Simon Lacoste-Julien. Implicit regularization of discrete gradient dynamics in linear neural networks. *Advances in Neural Information Processing Systems*, 32, 2019.

[104] Daniel Gissin, Shai Shalev-Shwartz, and Amit Daniely. The implicit bias of depth: How incremental learning drives generalization. *arXiv preprint arXiv:1909.12051*, 2019.

[105] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pages 249–256. JMLR Workshop and Conference Proceedings, 2010.

[106] Sebastian Goldt, Madhu Advani, Andrew M Saxe, Florent Krzakala, and Lenka Zdeborová. Dynamics of stochastic gradient descent for two-layer neural networks in the teacher-student setup. *Advances in neural information processing systems*, 32, 2019.

[107] Sebastian Goldt, Marc Mézard, Florent Krzakala, and Lenka Zdeborová. Modeling the influence of data structure on learning in neural networks: The hidden manifold model. *Physical Review X*, 10(4):041044, 2020.

[108] Rubén González-Sendino, Emilio Serrano, Javier Bajo, and Paulo Novais. A review of bias and fairness in artificial intelligence. *International Journal of Interactive Multimedia and Artificial Intelligence*, 2023.

[109] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

[110] Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013.

[111] Suriya Gunasekar, Jason Lee, Daniel Soudry, and Nathan Srebro. Characterizing implicit bias in terms of optimization geometry. In *International Conference on Machine Learning*, pages 1832–1841. PMLR, 2018.

[112] Suriya Gunasekar, Jason D Lee, Daniel Soudry, and Nati Srebro. Implicit bias of gradient descent on linear convolutional networks. *Advances in Neural Information Processing Systems*, 31, 2018.

[113] Suriya Gunasekar, Blake Woodworth, and Nathan Srebro. Mirrorless mirror descent: A natural derivation of mirror descent. In *International Conference on Artificial Intelligence and Statistics*, pages 2305–2313. PMLR, 2021.

[114] Torkel Hafting, Marianne Fyhn, Sturla Molden, May-Britt Moser, and Edvard I Moser. Microstructure of a spatial map in the entorhinal cortex. *Nature*, 436 (7052):801–806, 2005.

[115] Alon Halevy, Peter Norvig, and Fernando Pereira. The unreasonable effectiveness of data. *IEEE intelligent systems*, 24(2):8–12, 2009.

[116] James V Haxby, M Ida Gobbini, Maura L Furey, Alumit Ishai, Jennifer L Schouten, and Pietro Pietrini. Distributed and overlapping representations of faces and objects in ventral temporal cortex. *Science*, 293(5539):2425–2430, 2001.

[117] Soufiane Hayou, Nikhil Ghosh, and Bin Yu. The impact of initialization on lora finetuning dynamics. *arXiv preprint arXiv:2406.08447*, 2024.

[118] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *2015 IEEE International Conference on Computer Vision (ICCV)*, 12 2015. doi: 10.1109/iccv.2015.123. URL `https://ieeexplore.ieee.org/document/7410480/`.

[119] Christiaan Heij, André CM Ran, and Freek Van Schagen. *Introduction to mathematical systems theory*. Springer, 2007.

[120] Irina Higgins, Loic Matthey, Arka Pal, Christopher P Burgess, Xavier Glorot, Matthew M Botvinick, Shakir Mohamed, and Alexander Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. *ICLR (Poster)*, 3, 2017.

[121] Eleanor Holton, Lukas Braun, Jessica Thompson, Jan Grohn, and Christopher Summerfield. Humans and neural networks show similar patterns of transfer and interference during continual learning. *OSF*, 2025.

[122] John J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

[123] John J Hopfield. Hopfield network. *Scholarpedia*, 2(5):1977, 2007.

[124] Edward J Hu, yelong shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of

large language models. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=nZeVKeeFYf9`.

[125] Dongsung Huh. Curvature-corrected learning dynamics in deep neural networks. In *International Conference on Machine Learning*, pages 4552–4560. PMLR, 2020.

[126] Minyoung Huh, Hossein Mobahi, Richard Zhang, Brian Cheung, Pulkit Agrawal, and Phillip Isola. The low-rank simplicity bias in deep networks. *arXiv preprint arXiv:2103.10427*, 2021.

[127] Christiaan Huygens. 1673 horologium oscillatorium, sive, de motu pendulorum ad horologia aptato demonstrationes geometricae. *Paris: Apud F. Muguet*, 1966.

[128] Alumit Ishai, Leslie G Ungerleider, Alex Martin, and James V Haxby. The representation of objects in the human occipital and temporal cortex. *Journal of cognitive neuroscience*, 12(Supplement 2):35–51, 2000.

[129] Alicia Izquierdo, Jonathan L Brigman, Anna K Radke, Peter H Rudebeck, and Andrew Holmes. The neural basis of reversal learning: An updated perspective. *Neuroscience*, 345:12–26, 2017.

[130] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

[131] Arthur Jacot, François Ged, Berfin Şimşek, Clément Hongler, and Franck Gabriel. Saddle-to-saddle dynamics in deep linear networks: Small initialization training, symmetry, and sparsity. *arXiv preprint arXiv:2106.15933*, 2021.

[132] Anchit Jain, Rozhin Nobahari, Aristide Baratin, and Stefano Sarao Mannelli. Bias in motion: Theoretical insights into the dynamics of bias in sgd training. *arXiv preprint arXiv:2405.18296*, 2024.

[133] Anthony I Jang, Vincent D Costa, Peter H Rudebeck, Yogita Chudasama, Elisabeth A Murray, and Bruno B Averbeck. The role of frontal cortical and medial-temporal lobe brain areas in learning a bayesian prior belief on reversals. *Journal of Neuroscience*, 35(33):11751–11760, 2015.

[134] Devon Jarvis, Richard Klein, Benjamin Rosman, and Andrew M Saxe. On the specialization of neural modules. *arXiv preprint arXiv:2409.14981*, 2024.

[135] Devon Jarvis, Sebastian Lee, Clémentine Carla Juliette Dominé, Andrew M Saxe, and Stefano Sarao Mannelli. A theory of initialisation's impact on specialisation. In *NeurIPS 2024 Workshop on Mathematics of Modern Machine Learning*, 2024.

[136] Khurram Javed and Martha White. Meta-learning representations for continual learning. In *Advances in Neural Information Processing Systems*, pages 1820–1830, 2019.

[137] Ziwei Ji and Matus Telgarsky. Gradient descent aligns the layers of deep linear networks. *arXiv preprint arXiv:1810.02032*, 2018.

[138] WJ Johnston and S Fusi. Abstract representations emerge naturally in neural networks trained to perform multiple tasks. nat commun. 2023; 14: 1040, 2023.

[139] D Kaiser. The sacred, spherical cows of physics. *Nautilus Quaterly*, 2014.

[140] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International conference on machine learning*, pages 5156–5165. PMLR, 2020.

[141] Alexander JE Kell and Josh H McDermott. Deep neural network models of sensory systems: windows onto the role of task constraints. *Current opinion in neurobiology*, 55:121–132, 2019.

[142] Henry J Kelley. Gradient theory of optimal flight paths. *Ars Journal*, 30(10): 947–954, 1960.

[143] Mikail Khona and Ila Rani Fiete. Attractor and integrator networks in the brain. *Nature Reviews Neuroscience*, 23:744 – 766, 2021.

[144] Tim C Kietzmann, Patrick McClure, and Nikolaus Kriegeskorte. Deep neural networks in computational neuroscience. *BioRxiv*, page 133504, 2017.

[145] Hyunjik Kim and Andriy Mnih. Disentangling by factorising, 2019. URL `https://arxiv.org/abs/1802.05983`.

[146] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[147] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014. URL `http://arxiv.org/abs/1312.6114`.

[148] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13): 3521–3526, 2017.

[149] Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *International conference on machine learning*, pages 3519–3529. PMLR, 2019.

[150] Nikolaus Kriegeskorte. Deep neural networks: a new framework for modeling biological vision and brain information processing. *Annual review of vision science*, 1(1):417–446, 2015.

[151] Nikolaus Kriegeskorte, Marieke Mur, and Peter A Bandettini. Representational similarity analysis-connecting the branches of systems neuroscience. *Frontiers in systems neuroscience*, page 4, 2008.

[152] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[153] Jonas Kubilius, Martin Schrimpf, Kohitij Kar, Rishi Rajalingham, Ha Hong, Najib Majaj, Elias Issa, Pouya Bashivan, Jonathan Prescott-Roy, Kailyn Schmidt, et al. Brain-like object recognition with high-performing shallow recurrent anns. *Advances in neural information processing systems*, 32, 2019.

[154] Abhishek Kumar, Prasanna Sattigeri, and Avinash Balakrishnan. Variational inference of disentangled latent concepts from unlabeled observations, 2018. URL `https://arxiv.org/abs/1711.00848`.

[155] Tanishq Kumar, Blake Bordelon, Samuel J Gershman, and Cengiz Pehlevan. Grokking as the transition from lazy to rich training dynamics. *arXiv preprint arXiv:2310.06110*, 2023.

[156] Tanishq Kumar, Blake Bordelon, Cengiz Pehlevan, Venkatesh N Murthy, and Samuel J Gershman. Do mice grok? glimpses of hidden progress during overtraining in sensory cortex. *arXiv preprint arXiv:2411.03541*, 2024.

[157] Daniel Kunin, Atsushi Yamamura, Chao Ma, and Surya Ganguli. The asymmetric maximum margin bias of quasi-homogeneous neural networks. *arXiv preprint arXiv:2210.03820*, 2022.

[158] Daniel Kunin, Allan Raventós, Clémentine Dominé, Feng Chen, David Klindt, Andrew Saxe, and Surya Ganguli. Get rich quick: exact solutions reveal how unbalanced initializations promote rapid feature learning. *Advances in Neural Information Processing Systems*, 37:81157–81203, 2025.

[159] Aarre Laakso and Garrison Cottrell. Content and cluster analysis: assessing representational similarity in neural systems. *Philosophical psychology*, 13 (1):47–76, 2000.

[160] Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and brain sciences*, 40:e253, 2017.

[161] Andrew K Lampinen and Surya Ganguli. An analytic theory of generalization dynamics and transfer learning in deep linear networks. *arXiv preprint arXiv:1809.10374*, 2018.

[162] Janne K Lappalainen, Fabian D Tschopp, Sridhama Prakhya, Mason McGill, Aljoscha Nern, Kazunori Shinomiya, Shin-ya Takemura, Eyal Gruntman, Jakob H Macke, and Srinivas C Turaga. Connectome-constrained networks predict neural activity across the fly visual system. *Nature*, pages 1–9, 2024.

[163] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[164] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. URL `https://hal.science/hal-03926082/document`.

[165] Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *Advances in neural information processing systems*, 32, 2019.

[166] Jin Hwa Lee, Stefano Sarao Mannelli, and Andrew Saxe. Why do animals need shaping? a theory of task composition and curriculum learning. *Forty-first International Conference on Machine Learning, ICML 2024, Vienna,*

Austria, July 21-27, 2024, 2024. URL `https://openreview.net/forum?id=SODPCE7tt4`.

[167] Sebastian Lee, Sebastian Goldt, and Andrew Saxe. Continual learning in the teacher-student setup: Impact of task similarity. In *International Conference on Machine Learning*, pages 6109–6119. PMLR, 2021.

[168] Sebastian Lee, Stefano Sarao Mannelli, Claudia Clopath, Sebastian Goldt, and Andrew Saxe. Maslow's hammer for catastrophic forgetting: Node re-use vs node activation. *arXiv preprint arXiv:2205.09029*, 2022.

[169] Michael Lepori, Thomas Serre, and Ellie Pavlick. Break it down: Evidence for structural compositionality in neural networks. *Advances in Neural Information Processing Systems*, 36:42623–42660, 2023.

[170] Zhiyuan Li, Yuping Luo, and Kaifeng Lyu. Towards resolving the implicit bias of gradient descent for matrix factorization: Greedy low-rank learning. *arXiv preprint arXiv:2012.09839*, 2020.

[171] Zhiyuan Li, Tianhao Wang, Jason D Lee, and Sanjeev Arora. Implicit bias of gradient descent on reparametrized models: On equivalence to mirror descent. *Advances in Neural Information Processing Systems*, 35:34626–34640, 2022.

[172] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017.

[173] Jack W Lindsey and Samuel Lippl. Implicit regularization of multi-task learning and finetuning in overparameterized neural networks. *arXiv preprint arXiv:2310.02396*, 2023.

[174] Samuel Lippl and Jack Lindsey. Inductive biases of multi-task learning and finetuning: multiple regimes of feature reuse. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[175] Samuel Lippl, Kenneth Kay, Greg Jensen, Vincent P Ferrera, and LF Abbott. A mathematical theory of relational generalization in transitive inference. *Proceedings of the National Academy of Sciences*, 121(28):e2314511121, 2024.

[176] Yuhan Helena Liu, Aristide Baratin, Jonathan Cornford, Stefan Mihalas, Eric Shea-Brown, and Guillaume Lajoie. How connectivity structure shapes rich and lazy learning in neural circuits. *ArXiv*, pages arXiv–2310, 2024.

[177] Francesco Locatello, Stefan Bauer, Mario Lucic, Gunnar Raetsch, Sylvain Gelly, Bernhard Schölkopf, and Olivier Bachem. Challenging common assumptions in the unsupervised learning of disentangled representations. In *international conference on machine learning*, pages 4114–4124. PMLR, 2019.

[178] Francesco Locatello, Ben Poole, Gunnar Rätsch, Bernhard Schölkopf, Olivier Bachem, and Michael Tschannen. Weakly-supervised disentanglement without compromises, 2020. URL `https://arxiv.org/abs/2002.02886`.

[179] Leon Lufkin, Andrew M. Saxe, and Erin Grant. Nonlinear dynamics of localization in neural receptive fields, 2025. URL `https://arxiv.org/abs/2501.17284`.

[180] Tao Luo, Zhi-Qin John Xu, Zheng Ma, and Yaoyu Zhang. Phase diagram for two-layer relu neural networks at infinite-width limit. *Journal of Machine Learning Research*, 22(71):1–47, 2021.

[181] Kaifeng Lyu and Jian Li. Gradient descent maximizes the margin of homogeneous neural networks. *arXiv preprint arXiv:1906.05890*, 2019.

[182] Kaifeng Lyu, Jikai Jin, Zhiyuan Li, Simon Shaolei Du, Jason D Lee, and Wei Hu. Dichotomy of early and late phase implicit biases can provably induce grokking. In *The Twelfth International Conference on Learning Representations*, 2023.

[183] Sadhika Malladi, Alexander Wettig, Dingli Yu, Danqi Chen, and Sanjeev Arora. A kernel-based view of language model fine-tuning. In *International Conference on Machine Learning*, pages 23610–23641. PMLR, 2023.

[184] Valerio Mante, David Sussillo, Krishna V. Shenoy, and William T. Newsome. Context-dependent computation by recurrent dynamics in prefrontal cortex. *Nature*, 503:78 – 84, 2013.

[185] Sibylle Marcotte, Remi Gribonval, and Gabriel Peyré. Abide by the law and follow the flow: conservation laws for gradient flows, 12 2023. URL `https://proceedings.neurips.cc/paper_files/paper/2023/hash/c7bee9b76be21146fd592fc2b46614d5-Abstract-Conference.html`.

[186] Alex Martin. The representation of object concepts in the brain. *Annu. Rev. Psychol.*, 58(1):25–45, 2007.

[187] Nicolas Y. Masse, Guangyu Robert Yang, H. Francis Song, Xiao-Jing Wang, and David J. Freedman. Circuit mechanisms for the maintenance and manipulation of information in working memory. *Nature Neuroscience*, 22:1159 – 1167, 2018.

[188] Nicolas Y. Masse, Matthew C. Rosen, and David J. Freedman. Reevaluating the role of persistent neural activity in short-term memory. *Trends in Cognitive Sciences*, 24:242–258, 2020.

[189] Francesca Mastrogiuseppe and Srdjan Ostojic. Linking connectivity, dynamics, and computations in low-rank recurrent neural networks. *Neuron*, 99(3): 609–623, 2018.

[190] Loic Matthey, Irina Higgins, Demis Hassabis, and Alexander Lerchner. dsprites: Disentanglement testing sprites dataset, 2017.

[191] James L McClelland. Incorporating rapid neocortical learning of new schema-consistent information into complementary learning systems theory. *Journal of Experimental Psychology: General*, 142(4):1190, 2013.

[192] James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological review*, 102(3):419, 1995.

[193] Michael McCloskey and Neal J Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier, 1989.

[194] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5:115–133, 1943.

[195] Ninareh Mehrabi, Fred Morstatter, Nripsuta Saxena, Kristina Lerman, and Aram Galstyan. A survey on bias and fairness in machine learning. *ACM computing surveys (CSUR)*, 54(6):1–35, 2021.

[196] Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33):E7665–E7671, 2018.

[197] Dmytro Mishkin and Jiri Matas. All you need is a good init. *arXiv preprint arXiv:1511.06422*, 2015.

[198] Ari Morcos, Maithra Raghu, and Samy Bengio. Insights on representational similarity in neural networks with canonical correlation. *Advances in neural information processing systems*, 31, 2018.

[199] Edward Moroshko, Blake E Woodworth, Suriya Gunasekar, Jason D Lee, Nati Srebro, and Daniel Soudry. Implicit bias in deep linear classification: Initialization scale vs training accuracy. *Advances in neural information processing systems*, 33:22182–22193, 2020.

[200] Luca Moschella, Valentino Maiorca, Marco Fumero, Antonio Norelli,

Francesco Locatello, and Emanuele Rodolà. Relative representations enable zero-shot latent space communication, 2023.

[201] Samuel P Muscinelli, Mark J Wagner, and Ashok Litwin-Kumar. Optimal routing to cerebellum-like structures. *Nature neuroscience*, 26(9):1630–1641, 2023.

[202] Mor Shpigel Nacson, Suriya Gunasekar, Jason Lee, Nathan Srebro, and Daniel Soudry. Lexicographic and depth-sensitive margins in homogeneous and non-homogeneous deep models. In *International Conference on Machine Learning*, pages 4683–4692. PMLR, 2019.

[203] Yoonsoo Nam, Seok Hyeong Lee, Clementine Domine, Yea Chan Park, Charles London, Wonyl Choi, Niclas Goring, and Seungjai Lee. Position: Solve layerwise linear models first to understand neural dynamical phenomena (neural collapse, emergence, lazy/rich regime, and grokking). *arXiv preprint arXiv:2502.21009*, 2025.

[204] Neel Nanda, Lawrence Chan, Tom Lieberum, Jess Smith, and Jacob Steinhardt. Progress measures for grokking via mechanistic interpretability. *arXiv preprint arXiv:2301.05217*, 2023.

[205] Stephanie Nelli, Lukas Braun, Tsvetomira Dumbalska, Andrew Saxe, and Christopher Summerfield. Neural knowledge assembly in humans and neural networks. *Neuron*, 111(9):1504–1516, 2023.

[206] Ramon Nogueira, Chris C Rodgers, Randy M Bruno, and Stefano Fusi. The non-linear mixed representations in somatosensory cortex support simple and complex tasks. *BioRxiv*, pages 2021–02, 2021.

[207] Ramon Nogueira, Chris C Rodgers, Randy M Bruno, and Stefano Fusi. The geometry of cortical representations of touch in rodents. *Nature Neuroscience*, 26(2):239–250, 2023.

[208] Antonio Orvieto, Soham De, Caglar Gulcehre, Razvan Pascanu, and Samuel L Smith. Universality of linear recurrences followed by non-linear projections: Finite-width guarantees and benefits of complex eigenvalues. In *Forty-first International Conference on Machine Learning*, 2024.

[209] Keiron O'Shea and Ryan Nash. An introduction to convolutional neural networks, 2015. URL `https://arxiv.org/abs/1511.08458`.

[210] Srdjan Ostojic and Stefano Fusi. Computational role of structure in neural activity and connectivity. *Trends in Cognitive Sciences*, 2024.

[211] As paraphrased from Maslow. Ah, the psychology of science: A reconnaissance, 1966.

[212] German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019.

[213] Venkatesh Balavadhani Parthasarathy, Ahtsham Zafar, Aafaq Khan, and Arsalan Shahid. The ultimate guide to fine-tuning llms from basics to breakthroughs: An exhaustive review of technologies, research, best practices, applied research challenges and opportunities. *arXiv preprint arXiv:2408.13296*, 2024.

[214] Marco Pegoraro, Clémentine Dominé, Emanuele Rodolà, Petar Veličković, and Andreea Deac. Geometric epitope and paratope prediction. *Bioinformatics*, 40(7), 2024.

[215] Jeffrey Pennington, Samuel Schoenholz, and Surya Ganguli. Resurrecting the sigmoid in deep learning through dynamical isometry: theory and practice. *Advances in neural information processing systems*, 30, 2017.

[216] Hannah Pinson, Joeri Lenaerts, and Vincent Ginis. Linear cnns discover the statistical structure of the dataset using only the most dominant frequencies. In

International Conference on Machine Learning, pages 27876–27906. PMLR, 2023.

[217] Tomaso Poggio, Qianli Liao, Brando Miranda, Andrzej Banburski, Xavier Boix, and Jack Hidary. Theory iiib: Generalization in deep networks. *arXiv preprint arXiv:1806.11379*, 2018.

[218] Alethea Power, Yuri Burda, Harri Edwards, Igor Babuschkin, and Vedant Misra. Grokking: Generalization beyond overfitting on small algorithmic datasets. *arXiv preprint arXiv:2201.02177*, 2022.

[219] Alexandra Maria Proca, Clémentine Carla Juliette Dominé, Murray Shanahan, and Pedro AM Mediano. Learning dynamics in linear recurrent neural networks. In *Forty-second International Conference on Machine Learning*, 2024.

[220] William Qian, Jacob A Zavatone-Veth, Benjamin Samuel Ruben, and Cengiz Pehlevan. Partial observation can induce mechanistic mismatches in data-constrained rnns. In *The First Workshop on NeuroAI@ NeurIPS2024*, 2024.

[221] R Quian Quiroga, Leila Reddy, Gabriel Kreiman, Christof Koch, and Itzhak Fried. Invariant visual representation by single neurons in the human brain. *Nature*, 435(7045):1102–1107, 2005.

[222] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[223] Shahryar Rahnamayan and G Gary Wang. Toward effective initialization for large-scale search spaces. *Trans Syst*, 8(3):355–367, 2009.

[224] Rishi Rajalingham, Elias B Issa, Pouya Bashivan, Kohitij Kar, Kailyn Schmidt, and James J DiCarlo. Large-scale, high-resolution comparison of the core visual object recognition behavior of humans, monkeys, and state-of-the-art

deep artificial neural networks. *Journal of Neuroscience*, 38(33):7255–7269, 2018.

[225] Vinay V Ramasesh, Ethan Dyer, and Maithra Raghu. Anatomy of catastrophic forgetting: Hidden representations and task semantics. *arXiv preprint arXiv:2007.07400*, 2020.

[226] David Raposo, Matthew T Kaufman, and Anne K Churchland. A category-free neural population supports evolving demands during decision-making. *Nature neuroscience*, 17(12):1784–1792, 2014.

[227] Roger Ratcliff. Connectionist models of recognition memory: constraints imposed by learning and forgetting functions. *Psychological review*, 97(2): 285, 1990.

[228] Evan D. Remington, Seth W. Egger, Devika Narain, Jing Wang, and Mehrdad Jazayeri. A dynamical systems perspective on flexible motor timing. *Trends in Cognitive Sciences*, 22(10):938–952, 2018. ISSN 1364-6613. doi: https://doi.org/10.1016/j.tics.2018.07.010. Special Issue: Time in the Brain.

[229] Blake A Richards, Timothy P Lillicrap, Philippe Beaudoin, Yoshua Bengio, Rafal Bogacz, Amelia Christensen, Claudia Clopath, Rui Ponte Costa, Archy de Berker, Surya Ganguli, et al. A deep learning framework for neuroscience. *Nature neuroscience*, 22(11):1761–1770, 2019.

[230] Mattia Rigotti, Omri Barak, Melissa R Warden, Xiao-Jing Wang, Nathaniel D Daw, Earl K Miller, and Stefano Fusi. The importance of mixed selectivity in complex cognitive tasks. *Nature*, 497(7451):585–590, 2013.

[231] Grant Rotskoff and Eric Vanden-Eijnden. Parameters as interacting particles: long time convergence and asymptotic error scaling of neural networks. *Advances in neural information processing systems*, 31, 2018.

[232] Grant Rotskoff and Eric Vanden-Eijnden. Trainability and accuracy of artifi-

cial neural networks: An interacting particle system approach. *Communications on Pure and Applied Mathematics*, 75(9):1889–1935, 2022.

[233] Grant M Rotskoff and Eric Vanden-Eijnden. Neural networks as interacting particle systems: Asymptotic convexity of the loss landscape and universal scaling of the approximation error. *stat*, 1050:22, 2018.

[234] Noa Rubin, Inbar Seroussi, and Zohar Ringel. Grokking as a first order phase transition in two layer networks. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=3ROGsTX3IR`.

[235] Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*, 2016.

[236] David Saad and Sara Solla. Dynamics of on-line gradient descent learning for multilayer neural networks. *Advances in neural information processing systems*, 8, 1995.

[237] David Saad and Sara A Solla. Exact solution for on-line learning in multilayer neural networks. *Physical Review Letters*, 74(21):4337, 1995.

[238] David Saad and Sara A Solla. On-line learning in soft committee machines. *Physical Review E*, 52(4):4225, 1995.

[239] Luca Saglietti, Stefano Mannelli, and Andrew Saxe. An analytical theory of curriculum learning in teacher-student networks. *Advances in Neural Information Processing Systems*, 35:21113–21127, 2022.

[240] Kai Jappe Sandbrink, Jan Philipp Bauer, Alexandra Maria Proca, Andrew M Saxe, Christopher Summerfield, and Ali Hummos. Flexible task abstractions emerge in linear networks with fast and bounded units. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL `https://openreview.net/forum?id=AbTpJl7vN6`.

[241] Andrew Saxe, Stephanie Nelli, and Christopher Summerfield. If deep learning is the answer, then what is the question? *arXiv:2004.07580 [q-bio]*, 04 2020. URL `https://arxiv.org/abs/2004.07580`.

[242] Andrew Saxe, Shagun Sodhani, and Sam Jay Lewallen. The neural race reduction: Dynamics of abstraction in gated networks. In *International Conference on Machine Learning*, pages 19287–19309. PMLR, 2022.

[243] Andrew M. Saxe, James L. McClelland, and Surya Ganguli. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[244] Andrew M. Saxe, James L. McClelland, and Surya Ganguli. A mathematical theory of semantic development in deep neural networks. *Proceedings of the National Academy of Sciences*, 116:11537 – 11546, 2018. URL `https://api.semanticscholar.org/CorpusID:53024512`.

[245] Andrew M Saxe, Yamini Bansal, Joel Dapello, Madhu Advani, Artemy Kolchinsky, Brendan D Tracey, and David D Cox. On the information bottleneck theory of deep learning. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):124020, 2019.

[246] Andrew M. Saxe, James L. McClelland, and Surya Ganguli. A mathematical theory of semantic development in deep neural networks. *Proceedings of the National Academy of Sciences*, 116(23):11537–11546, 2019. doi: 10.1073/pnas.1820226116. URL `https://www.pnas.org/content/116/23/11537`.

[247] Andrew M Saxe, James L McClelland, and Surya Ganguli. A mathematical theory of semantic development in deep neural networks. *Proceedings of the National Academy of Sciences*, 116(23):11537–11546, 2019.

[248] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

[249] Martin Schrimpf, Jonas Kubilius, Ha Hong, Najib J Majaj, Rishi Rajalingham, Elias B Issa, Kohitij Kar, Pouya Bashivan, Jonathan Prescott-Roy, Franziska Geiger, et al. Brain-score: Which artificial neural network for object recognition is most brain-like? *BioRxiv*, page 407007, 2018.

[250] Martin Schrimpf, Jonas Kubilius, Ha Hong, Najib J Majaj, Rishi Rajalingham, Elias B Issa, Kohitij Kar, Pouya Bashivan, Jonathan Prescott-Roy, Franziska Geiger, et al. Brain-score: Which artificial neural network for object recognition is most brain-like? *BioRxiv*, page 407007, 2020.

[251] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588(7839):604–609, 2020.

[252] Friedrich Schuessler, Alexis Dubreuil, Francesca Mastrogiuseppe, Srdjan Ostojic, and Omri Barak. Dynamics of random recurrent networks with correlated low-rank structure. *Physical Review Research*, 2(1):013111, 2020.

[253] Friedrich Schuessler, Francesca Mastrogiuseppe, Alexis Dubreuil, Srdjan Ostojic, and Omri Barak. The interplay between randomness and structure during learning in rnns. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 13352–13362. Curran Associates, Inc., 2020.

[254] Anton Maximilian Schäfer and Hans Georg Zimmermann. Recurrent neural networks are universal approximators. *International Journal of Neural Systems*, 17(4):253–63, 2007.

[255] Hyunjune Sebastian Seung, Haim Sompolinsky, and Naftali Tishby. Statistical mechanics of learning from examples. *Physical review A*, 45(8):6056, 1992.

[256] Gal Shachaf, Alon Brutzkus, and Amir Globerson. A theoretical analysis of fine-tuning with linear teachers. *Advances in Neural Information Processing Systems*, 34, 2021.

[257] Lee Sharkey, Bilal Chughtai, Joshua Batson, Jack Lindsey, Jeff Wu, Lucius Bushnaq, Nicholas Goldowsky-Dill, Stefan Heimersheim, Alejandro Ortega, Joseph Bloom, et al. Open problems in mechanistic interpretability. *arXiv preprint arXiv:2501.16496*, 2025.

[258] James B Simon, Maksis Knutins, Liu Ziyin, Daniel Geisz, Abraham J Fetterman, and Joshua Albrecht. On the stepwise nature of self-supervised learning. In *International Conference on Machine Learning*, pages 31852–31876. PMLR, 2023.

[259] Fabian H Sinz, Xaq Pitkow, Jacob Reimer, Matthias Bethge, and Andreas S Tolias. Engineering a less artificial intelligence. *Neuron*, 103(6):967–979, 2019.

[260] Justin Sirignano and Konstantinos Spiliopoulos. Mean field analysis of neural networks: A law of large numbers. *SIAM Journal on Applied Mathematics*, 80(2):725–752, 2020.

[261] Shagun Sodhani, Mojtaba Faramarzi, Sanket Vaibhav Mehta, Pranshu Malviya, Mohamed Abdelsalam, Janarthanan Janarthanan, and Sarath Chandar. An introduction to lifelong supervised learning. *arXiv preprint arXiv:2207.04354*, 2022.

[262] Haim Sompolinsky, Andrea Crisanti, and Hans-Jurgen Sommers. Chaos in random neural networks. *Physical review letters*, 61(3):259, 1988.

[263] Haim Sompolinsky, Naftali Tishby, and H Sebastian Seung. Learning from examples in large neural networks. *Physical Review Letters*, 65(13):1683, 1990.

[264] Ben Sorscher, Gabriel Mel, Surya Ganguli, and Samuel Ocko. A unified theory for the origin of grid cells through the lens of pattern formation. *Advances in neural information processing systems*, 32, 2019.

[265] Daniel Soudry, Elad Hoffer, Mor Shpigel Nacson, Suriya Gunasekar, and Nathan Srebro. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research*, 19(1):2822–2878, 2018.

[266] Ilia Sucholutsky, Lukas Muttenthaler, Adrian Weller, Andi Peng, Andreea Bobu, Been Kim, Bradley C. Love, Erin Grant, Iris Groen, Jascha Achterberg, Joshua B. Tenenbaum, Katherine M. Collins, Katherine L. Hermann, Kerem Oktar, Klaus Greff, Martin N. Hebart, Nori Jacoby, Qiuyi Zhang, Raja Marjieh, Robert Geirhos, Sherol Chen, Simon Kornblith, Sunayana Rane, Talia Konkle, Thomas P. O'Connell, Thomas Unterthiner, Andrew K. Lampinen, Klaus-Robert Müller, Mariya Toneva, and Thomas L. Griffiths. Getting aligned on representational alignment, 2023.

[267] Steven C Suddarth and YL Kergosien. Rule-injection hints as a means of improving network performance and learning time. In *European association for signal processing workshop*, pages 120–129. Springer, 1990.

[268] Chen Sun, Abhinav Shrivastava, Saurabh Singh, and Abhinav Gupta. Revisiting unreasonable effectiveness of data in deep learning era. In *Proceedings of the IEEE international conference on computer vision*, pages 843–852, 2017.

[269] Chen Sun, Wannan Yang, Jared Martin, and Susumu Tonegawa. Hippocampal neurons represent events as transferable units of experience. *Nature neuroscience*, 23(5):651–663, 2020.

[270] Chuanqi Tan, Fuchun Sun, Tao Kong, Wenchang Zhang, Chao Yang, and Chunfang Liu. A survey on deep transfer learning. In *Artificial Neural Networks and Machine Learning–ICANN 2018: 27th International Conference on Artificial Neural Networks, Rhodes, Greece, October 4-7, 2018, Proceedings, Part III 27*, pages 270–279. Springer, 2018.

[271] Evelyn Tang, Marcelo G Mattar, Chad Giusti, David M Lydon-Staley, Sharon L Thompson-Schill, and Danielle S Bassett. Effective learning is

accompanied by high-dimensional and efficient representations of neural activity. *Nature neuroscience*, 22(6):1000–1009, 2019.

[272] Salma Tarmoun, Guilherme Franca, Benjamin D Haeffele, and Rene Vidal. Understanding the dynamics of gradient flow in overparameterized linear models. In *International Conference on Machine Learning*, pages 10153–10161. PMLR, 2021.

[273] Matthew E Taylor and Peter Stone. Transfer learning for reinforcement learning domains: A survey. *Journal of Machine Learning Research*, 10(7), 2009.

[274] Adly Templeton. *Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet.* Anthropic, 2024.

[275] Emmett Thompson, Lars Rollik, Benjamin Waked, Georgina Mills, Jasvin Kaur, Ben Geva, Rodrigo Carrasco-Davis, Tom George, Clementine Domine, William Dorrell, et al. Replay of procedural experience is independent of the hippocampus. *bioRxiv*, pages 2024–06, 2024.

[276] Sebastian Thrun and Lorien Pratt. *Learning to learn.* Springer Science & Business Media, 2012.

[277] Yuandong Tian. An analytical formula of population gradient for two-layered relu network and its applications in convergence and critical point analysis. In *International conference on machine learning*, pages 3404–3413. PMLR, 2017.

[278] Mariya Toneva, Alessandro Sordoni, Remi Tachet des Combes, Adam Trischler, Yoshua Bengio, and Geoffrey J Gordon. An empirical study of example forgetting during deep neural network learning. *arXiv preprint arXiv:1812.05159*, 2018.

[279] Nilesh Tripuraneni, Michael Jordan, and Chi Jin. On the theory of transfer

learning: The importance of task diversity. *Advances in Neural Information Processing Systems*, 33:7852–7862, 2020.

[280] Zhenfeng Tu, Santiago Aranguri, and Arthur Jacot. Mixed dynamics in linear networks: Unifying the lazy and active regimes. *arXiv preprint arXiv:2405.17580*, 2024.

[281] Elia Turner, Kabir Dabholkar, and Omri Barak. Charting and navigating the space of solutions for recurrent neural networks. In *Neural Information Processing Systems*, 2021.

[282] Kay Tye, Earl Miller, Felix Taschbach, Marcus Benna, Mattia Rigotti, and Stefano Fusi. Mixed selectivity: Cellular computations for complexity. *Neuron*, 112, 05 2024. doi: 10.1016/j.neuron.2024.04.017.

[283] Benton J Underwood. Interference and forgetting. *Psychological review*, 64 (1):49, 1957.

[284] Gal Vardi and Ohad Shamir. Implicit regularization in relu networks with the square loss. In *Conference on Learning Theory*, pages 4224–4258. PMLR, 2021.

[285] Aditya Vardhan Varre, Maria-Luiza Vladarean, Loucas Pillaud-Vivien, and Nicolas Flammarion. On the spectral bias of two-layer linear networks. *Advances in Neural Information Processing Systems*, 36, 2024.

[286] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[287] Rodrigo Veiga, Ludovic Stephan, Bruno Loureiro, Florent Krzakala, and Lenka Zdeborová. Phase diagram of stochastic gradient descent in high-dimensional two-layer neural networks. *Advances in Neural Information Processing Systems*, 35:23244–23255, 2022.

[288] Elena Voita, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 5797–5808. Association for Computational Linguistics, 2019. doi: 10.18653/V1/P19-1580. URL `https://doi.org/10.18653/v1/p19-1580`.

[289] Saurabh Vyas, Matthew D. Golub, David Sussillo, and Krishna V. Shenoy. Computation through neural population dynamics. *Annual review of neuroscience*, 43:249–275, 2020.

[290] Jing Wang, Devika Narain, Eghbal A. Hosseini, and Mehrdad Jazayeri. Flexible timing by temporal scaling of cortical responses. *Nature neuroscience*, 21: 102 – 110, 2017.

[291] Timothy LH Watkin, Albrecht Rau, and Michael Biehl. The statistical mechanics of learning a rule. *Reviews of Modern Physics*, 65(2):499, 1993.

[292] James CR Whittington and Rafal Bogacz. Theories of error back-propagation in the brain. *Trends in cognitive sciences*, 23(3):235–250, 2019.

[293] James CR Whittington, Joseph Warren, and Timothy EJ Behrens. Relating transformers to models and neural representations of the hippocampal formation. *arXiv preprint arXiv:2112.04035*, 2021.

[294] Blake Woodworth, Suriya Gunasekar, Jason D Lee, Edward Moroshko, Pedro Savarese, Itay Golan, Daniel Soudry, and Nathan Srebro. Kernel and rich regimes in overparametrized models. In *Conference on Learning Theory*, pages 3635–3673. PMLR, 2020.

[295] Marjorie Xie, Samuel P Muscinelli, Kameron Decker Harris, and Ashok Litwin-Kumar. Task-dependent optimal representations for cerebellar learning. *Elife*, 12:e82914, 2023.

[296] Xiangxiang Xu and Lizhong Zheng. Neural feature learning in function space *. *Journal of Machine Learning Research*, 25:1–76, 2024. URL `https://jmlr.org/papers/volume25/23-1202/23-1202.pdf`.

[297] Yizhou Xu and Liu Ziyin. When does feature learning happen? perspective from an analytically solvable model. *arXiv preprint arXiv:2401.07085*, 2024.

[298] Daniel L Yamins, Ha Hong, Charles Cadieu, and James J DiCarlo. Hierarchical modular optimization of convolutional networks achieves representations similar to macaque it and human ventral stream. *Advances in neural information processing systems*, 26, 2013.

[299] Daniel LK Yamins and James J DiCarlo. Using goal-driven deep learning models to understand sensory cortex. *Nature neuroscience*, 19(3):356–365, 2016.

[300] Greg Yang. Tensor programs ii: Neural tangent kernel for any architecture. *arXiv preprint arXiv:2006.14548*, 2020.

[301] Greg Yang and Edward J Hu. Feature learning in infinite-width neural networks. *arXiv preprint arXiv:2011.14522*, 2020.

[302] Thomas Yerxa, Yilun Kuang, Eero Simoncelli, and SueYeon Chung. Learning efficient coding of natural images with maximum manifold capacity representations. *Advances in Neural Information Processing Systems*, 36: 24103–24128, 2023.

[303] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? *Advances in neural information processing systems*, 27, 2014.

[304] Lenka Zdeborová and Florent Krzakala. Statistical physics of inference: Thresholds and algorithms. *Advances in Physics*, 65(5):453–552, 2016.

[305] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014: 13th European Conference,*

Zurich, Switzerland, September 6-12, 2014, Proceedings, Part I 13, pages 818–833. Springer, 2014.

[306] Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International Conference on Machine Learning*, pages 3987–3995. PMLR, 2017.

[307] Xiao Zhang, Yaodong Yu, Lingxiao Wang, and Quanquan Gu. Learning one-hidden-layer relu networks via gradient descent. In *The 22nd international conference on artificial intelligence and statistics*, pages 1524–1534. PMLR, 2019.

[308] Yedi Zhang, Andrew Saxe, and Peter E Latham. When are bias-free relu networks like linear networks? *arXiv preprint arXiv:2406.12615*, 2024.

[309] Zhenglong Zhou and Chaz Firestone. Humans can decipher adversarial images. *Nature communications*, 10(1):1334, 2019.

[310] Fuzhen Zhuang, Zhiyuan Qi, Keyu Duan, Dongbo Xi, Yongchun Zhu, Hengshu Zhu, Hui Xiong, and Qing He. A comprehensive survey on transfer learning. *Proceedings of the IEEE*, 109(1):43–76, 2020.

[311] Matthieu Zimmer, Paolo Viappiani, and Paul Weng. Teacher-student framework: a reinforcement learning approach. In *AAMAS Workshop autonomous robots and multirobot systems*, 2014.

[312] Liu Ziyin, Botao Li, and Xiangming Meng. Exact solutions of a deep linear network. *Advances in Neural Information Processing Systems*, 35:24446–24458, 2022.

[313] Difan Zou, Yuan Cao, Dongruo Zhou, and Quanquan Gu. Gradient descent optimizes over-parameterized deep relu networks. *Machine learning*, 109: 467–492, 2020.

[314] Nicolas Zucchet and Antonio Orvieto. Recurrent neural networks: vanishing and exploding gradients are not the end of the story. *Neural Information Processing Systems*, abs/2405.21064, 2024.