The Astrophysical Journal Supplement Series, 249:5 (13pp), 2020 July https://doi.org/10.3847/1538-4365/ab917f

SPECULATOR: Emulating Stellar Population Synthesis for Fast and Accurate Galaxy Spectra and Photometry

Justin Alsing[1], Hiranya Peiris[1,2], Joel Leja[3,9], ChangHoon Hahn[4,5], Rita Tojeiro[6], Daniel Mortlock[1,7], Boris Leistedt[8,10], Benjamin D. Johnson[3], and Charlie Conroy[3]
[1] Oskar Klein Centre for Cosmoparticle Physics, Department of Physics, Stockholm University, Stockholm SE-106 91, Sweden; justin.alsing@fysik.su.se
[2] Department of Physics and Astronomy, University College London, Gower Street, London, WC1E 6BT, UK
[3] Harvard-Smithsonian Center for Astrophysics, 60 Garden Street, Cambridge, MA 02138, USA
[4] Lawrence Berkeley National Laboratory, 1 Cyclotron Road, Berkeley, CA 94720, USA
[5] Berkeley Center for Cosmological Physics, University of California, Berkeley, CA 94720, USA
[6] School of Physics and Astronomy, University of St Andrews, North Haugh, St Andrews, KY16 9SS, UK
[7] Department of Physics, Imperial College London, Blackett Laboratory, Prince Consort Road, London SW7 2AZ, UK
[8] Center for Cosmology and Particle Physics, Department of Physics, New York University, New York, NY, USA
Received 2019 December 2; revised 2020 March 24; accepted 2020 April 3; published 2020 June 25

Abstract

We present SPECULATOR—a fast, accurate, and flexible framework for emulating stellar population synthesis (SPS) models for predicting galaxy spectra and photometry. For emulating spectra, we use a principal component analysis to construct a set of basis functions and neural networks to learn the basis coefficients as a function of the SPS model parameters. For photometry, we parameterize the magnitudes (for the filters of interest) as a function of SPS parameters by a neural network. The resulting emulators are able to predict spectra and photometry under both simple and complicated SPS model parameterizations to percent-level accuracy, giving a factor of 10^3–10^4 speedup over direct SPS computation. They have readily computable derivatives, making them amenable to gradient-based inference and optimization methods. The emulators are also straightforward to call from a GPU, giving an additional order of magnitude speedup. Rapid SPS computations delivered by emulation offers a massive reduction in the computational resources required to infer the physical properties of galaxies from observed spectra or photometry and simulate galaxy populations under SPS models, while maintaining the accuracy required for a range of applications.

Unified Astronomy Thesaurus concepts: Galaxies (573); Neural networks (1933); Galaxy photometry (611)

1. Introduction

Inferring the physical properties of galaxies from observations of the spectral energy distribution (SED) of their emitted light is one of the cornerstones of modern extragalactic astronomy. At the heart of this endeavor is stellar population synthesis (SPS): predictive models for galaxy SEDs that fold together the initial stellar mass function, star formation and metallicity enrichment histories, stellar evolution calculations and stellar spectral libraries, phenomenological dust and gas models, black hole activity etc., to predict the spectrum of a galaxy given some input physical parameters associated with each model component. SPS modeling has a rich history, with a plethora of parameterizations of varying complexity available (see Conroy 2013 and references therein).

The computational bottleneck in both inferring galaxy properties from observations and simulating catalogs under SPS models is running the SPS models themselves. Forward-simulating upcoming Stage IV galaxy surveys will demand $\sim10^{10}$ SPS evaluations per catalog simulation. For data analysis, inferring[11] of the order of 10 SPS model parameters for a single galaxy (given some photometric or spectroscopic data) typically requires $\sim10^5$–10^6 SPS model evaluations. If inference is then to be performed for a large sample of galaxies, the number of SPS evaluations and associated computational demands quickly becomes prohibitive. For recent context, Leja et al. (2019) analyzed $\sim6 \times 10^4$ galaxies under a 14 parameter SPS model, with a total cost of 1.5 million CPU hours.[12] With upcoming surveys such as the Dark Energy Spectroscopic Instrument (DESI; Levi et al. 2013; DESI Collaboration et al. 2016a, 2016b) posing the challenge of analyzing millions of galaxy spectra, the need to address the bottleneck posed by SPS is clear and urgent.

There are two principal ways of reducing the cost of inference and simulation under SPS models: speeding up individual SPS computations and (in the case of inference) reducing the number of SPS computations required to obtain robust inferences per galaxy. In this paper, we present neural network emulators for SPS spectra and photometry that gain leverage on both fronts. For galaxy spectra, our emulation framework uses a principal component analysis (PCA) to construct a basis for galaxy SEDs and then trains a neural network on a set of generated SPS spectra to learn the PCA basis coefficients as a function of the SPS model parameters. For photometry, we train a neural network to learn the magnitudes directly (for some set of bandpasses) as a function of the SPS parameters. The result in both cases is a compact neural network representation of the SPS model that is both fast to evaluate, accurate, and has analytic and readily computable derivatives, thus making it amenable to efficient gradient-based optimization and inference methods (e.g., Hamiltonian Monte Carlo sampling). Furthermore, calling the emulators from a GPU is straightforward, enabling an additional order of

[9] NSF Fellow.
[10] NASA Einstein Fellow.
[11] E.g., Markov Chain Monte Carlo sampling.

[12] For added context, the CPU time for the Leja et al. (2019) analysis would cost around $20,000 USD from Amazon Web Services (estimated in 2019).

magnitude speedup when evaluating many SPS models in parallel.

We demonstrate and validate the emulator on two SPS models[13]: one relatively simple eight parameter model targeting upcoming DESI observations (for which we emulate spectra) and the more flexible 14 parameter Prospector-α model from the recent Leja et al. (2019) analysis (for which we emulate both spectra and photometry). For both models, we show that the emulator is able to deliver percent-level accuracy over broad parameter prior and wavelength ranges and gives a factor of $\sim 10^3$–10^4 speedup over direct SPS model evaluation. T use of gradient-based inference methods enabled by the emulators will provide further reductions in the cost of inference under SPS models.

The structure of this paper is as follows: in Section 2 we outline the emulation framework. In Sections 3 and 4 we validate the spectrum emulator on two SPS model parameterizations. In Section 5 we validate the photometry emulator for the Prospector-α model. We discuss the implications for current and future studies in Section 6.

2. SPECULATOR: Emulating SPS

In this section we describe the framework developed for fast emulation of SPS spectra (Section 2.2) and photometry (Section 2.3). Some background knowledge of PCA and neural networks is assumed in this section; see e.g., Bishop (2006) for a comprehensive and pedagogical review. For previous work on representing spectra as interpolations over PCA bases, see Han & Han (2014), Czekala et al. (2015), and Kalmbach & Connolly (2017).

2.1. Notation

We will denote galaxy SEDs by $l(\lambda;\, \boldsymbol{\theta}) \equiv l_\lambda$ (luminosity per unit wavelength) and log SEDs by $L_\lambda \equiv \ln l_\lambda$ for wavelength λ and SPS model parameters $\boldsymbol{\theta}$. Photometric fluxes, denoted by $f_b(\boldsymbol{\theta})$, for a given bandpass b with filter $W_b(\lambda)$ and SPS parameters $\boldsymbol{\theta}$ are given by

$$f_b(\boldsymbol{\theta}) = \frac{1}{g^{\mathrm{AB}} 4\pi (1+z) d_L^2(z)} \int_0^\infty l(\lambda/(1+z);\, \boldsymbol{\theta}) W_b(\lambda) d\lambda, \tag{1}$$

where g^{AB} is the AB flux normalization, $d_L(z)$ is the luminosity distance for redshift z, and the filter is assumed to be normalized to unity, $\int W_b(\lambda) d\lambda = 1$. The associated apparent magnitudes are denoted by $m_b(\boldsymbol{\theta})$.

The goal of emulation is to find an efficient representation for the galaxy spectra $l_\lambda(\boldsymbol{\theta})$ or photometry $\{m_b(\boldsymbol{\theta})\}$ as a function of the SPS model parameters that is as fast as possible to evaluate, while maintaining accuracy.

2.2. Emulation of Galaxy Spectra

2.2.1. Parameterization Considerations

There are a couple of simplifications to the SED-emulation problem setup that will make emulation significantly easier.

We will emulate the rest-frame SEDs only, redshifting (analytically) afterward as needed. This is motivated by the fact that emulator is contingent on finding a compact PCA basis for galaxy SEDs; constructing this basis is greatly simplified when working within the rest frame only, i.e., without requiring that the basis can capture arbitrary stretches in wavelength. Meanwhile, emulating rest-frame SEDs only does not reduce functionality, since redshifted spectra can be obtained straightforwardly (and exactly) from the rest-frame SEDs.

Redshifting involves three transformations on the emulated rest-frame SEDs: a stretch by $\lambda \to \lambda/(1+z)$, rescaling by $[(1+z)d_L(z)^2]^{-1}$, and adjusting the age of the universe at the lookback time for a given redshift, $t_{\mathrm{age}}(z)$, so that the age of the stellar population is consistent with that lookback time. Therefore, $t_{\mathrm{age}}(z)$ must be included in the list of SPS parameters $\boldsymbol{\theta}$.

Similarly, we fix the total stellar mass, M, for the emulated spectra to $1\, M_\odot$ and scale the mass analytically afterward as required (the total stellar mass that formed M enters as a simple normalization of the SED). Hence, a galaxy spectrum for a given redshift z, total stellar mass formed M, and SPS model parameters $\boldsymbol{\theta}$ can be obtained from the corresponding emulated rest-frame unit stellar mass SED $l(\lambda;\, \boldsymbol{\theta})$ as

$$l(\lambda;\, \boldsymbol{\theta}, M, z) \to l(\lambda/(1+z);\, \boldsymbol{\theta})|_{t_{\mathrm{age}}(z)} \frac{1}{(1+z) d_{\mathrm{L}}(z)^2} M. \tag{2}$$

2.2.2. PCA Neural Network Emulator Framework

A schematic overview of the PCA network emulator framework described below is given in Figure 1 for reference throughout this section.

To build an emulator for a given SPS model parameterization, we begin by generating a training set of N_{train} galaxy SEDs $\{(L_\lambda, \boldsymbol{\theta})_1, (L_\lambda, \boldsymbol{\theta})_2, \ldots, (L_\lambda, \boldsymbol{\theta})_{N_{\mathrm{train}}}\}$ under the target SPS model, by drawing SPS parameters from the prior and computing the associated SEDs.

From this training set, we construct a basis $\{q_{\lambda,i}\}$ for the SEDs by performing a PCA decomposition of the training spectra and by taking the first N_{pca} principal components as basis vectors. The number of PCA components retained is chosen such that the resulting PCA basis is comfortably able to recover the model SEDs at the desired accuracy (i.e., $\ll 1\%$ if we want to ensure that the errors associated with the PCA basis are a small fraction of the total error budget).

With the PCA basis $\{q_{\lambda,i}\}$ in hand, we model the (log) SED as a linear combination of the PCA basis functions:

$$L_\lambda(\boldsymbol{\theta}) = \sum_{i=1}^{N_{\mathrm{pca}}} \alpha_i(\boldsymbol{\theta}) q_{\lambda,i}, \tag{3}$$

where the vector of coefficients $\boldsymbol{\alpha}(\boldsymbol{\theta})$ are some unknown (nonlinear) functions of the SPS parameters $\boldsymbol{\theta}$. The remaining step, then, is to learn some convenient parametric model $\hat{\boldsymbol{\alpha}}(\boldsymbol{\theta};\, \boldsymbol{w})$ (with parameters $\boldsymbol{w}$) for the basis coefficients $\boldsymbol{\alpha}(\boldsymbol{\theta})$ as a function of the SPS parameters.

We parameterize the basis coefficients as a function of the model parameters by a dense, fully connected neural network with n hidden layers, with $\{h_1, h_2, \ldots, h_n\}$ hidden units, and nonlinear activation functions $\{a_1, a_2, \ldots, a_n\}$,

[13] Implemented with the SPS code FSPS (Conroy et al. 2009; Conroy & Gunn 2010) with python bindings PYTHON-FSPS (Foreman-Mackey et al. 2014).

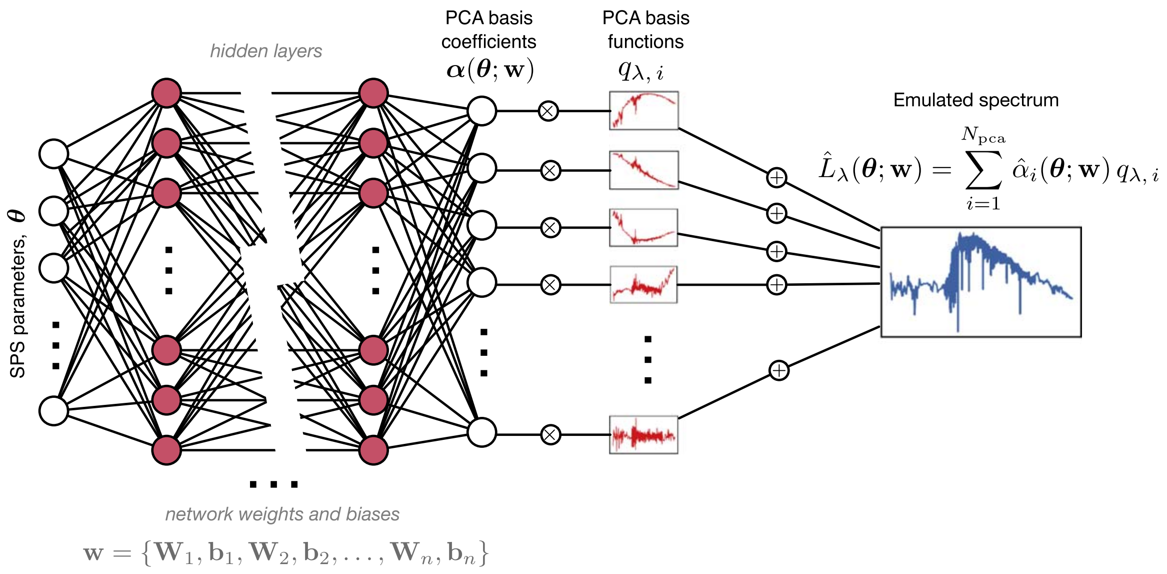
hidden layers
PCA basis coefficients
$\boldsymbol{\alpha}(\boldsymbol{\theta}; \mathbf{w})$
PCA basis functions
$q_{\lambda,\,i}$
Emulated spectrum
$\hat{L}_\lambda(\boldsymbol{\theta}; \mathbf{w}) = \sum_{i=1}^{N_{\rm pca}} \hat{\alpha}_i(\boldsymbol{\theta}; \mathbf{w})\, q_{\lambda,\,i}$
SPS parameters, $\boldsymbol{\theta}$
network weights and biases
$\mathbf{w} = \{\mathbf{W}_1, \mathbf{b}_1, \mathbf{W}_2, \mathbf{b}_2, \ldots, \mathbf{W}_n, \mathbf{b}_n\}$

Figure 1. Schematic of the PCA neural network emulator setup. A dense neural network parameterizes the PCA basis coefficients as a function of the SPS model parameters (i.e., taking SPS parameters as an input and predicting the basis coefficients). These basis coefficients are then multiplied by their respective PCA basis functions and summed to give the predicted spectrum.

respectively, i.e.,

$$\begin{aligned}\hat{\boldsymbol{\alpha}}(\boldsymbol{\theta};\, \boldsymbol{w}) &= a_n(\boldsymbol{W}_n \boldsymbol{y}_{n-1} + \boldsymbol{b}_n),\\ \boldsymbol{y}_{n-1} &= a_{n-1}(\boldsymbol{W}_{n-1}\boldsymbol{y}_{n-2} + \boldsymbol{b}_{n-1})\\ &\vdots\\ \boldsymbol{y}_1 &= a_1(\boldsymbol{W}_1\boldsymbol{\theta} + \boldsymbol{b}_1). \end{aligned} \tag{4}$$

The weight matrices and bias vectors for each network layer are denoted by $\boldsymbol{W}_k \in \mathbb{R}^{h_k \times h_{k-1}}$ and $\boldsymbol{b}_k \in \mathbb{R}^k$, so we use $\boldsymbol{w} = \{\boldsymbol{W}_k, \boldsymbol{b}_k\}$ as shorthand for the full set of weights and biases of the whole network, and $\boldsymbol{y}_k$ denotes the output from layer k.

Finally, to train the emulator, we optimize the network parameters $\boldsymbol{w}$ by minimizing the loss function:

$$-\ln U(\boldsymbol{w};\, \{\boldsymbol{\theta}, \boldsymbol{\alpha}\}) = \frac{1}{N_{\rm train}} \sum_{m=1}^{N_{\rm train}} |\boldsymbol{\alpha}_m - \hat{\boldsymbol{\alpha}}(\boldsymbol{\theta}_m;\, \boldsymbol{w})|^2, \tag{5}$$

where $\{\boldsymbol{\alpha}_m\}$ are the PCA basis coefficients for the SEDs $\{L_\lambda\}$ in the training set, and $\boldsymbol{\theta}_m$ is the corresponding SPS model parameters for those training set members. This loss function is just the mean square error between neural network predicted and true PCA basis coefficients over the training set.

The emulator model is succinctly summarized by

$$\hat{\boldsymbol{L}}(\boldsymbol{\theta}) = \boldsymbol{Q}\, \hat{\boldsymbol{\alpha}}(\boldsymbol{\theta};\, \boldsymbol{w}), \tag{6}$$

where $\hat{\boldsymbol{L}}(\boldsymbol{\theta}) = (\hat{L}_{\lambda,1}(\boldsymbol{\theta}), \hat{L}_{\lambda,2}(\boldsymbol{\theta}),..., \hat{L}_{\lambda,N_\lambda}(\boldsymbol{\theta}))$ is the emulated SED for parameters $\boldsymbol{\theta}$, $Q_{\lambda i} = q_{\lambda,i}$ is the set of basis functions, and $\hat{\boldsymbol{\alpha}}(\boldsymbol{\theta};\, \boldsymbol{w})$ is given by Equation (4). The neural network emulator is specified entirely by the set of matrices and nonlinear activation functions, $\{\boldsymbol{W}_k, \boldsymbol{b}_k, \boldsymbol{Q}, a_k\}$. Calculating an emulated SPS model spectrum using Equations (6) and (4) is hence reduced to a series of linear matrix operations and passes through simple nonlinear (e.g., tanh) activation functions. Furthermore, the neural network in Equation (4) is straightforwardly differentiable (by the chain rule), so derivatives of the model spectra with respect to the SPS parameters are readily available. We highlight that implementation of the trained emulator using Equations (4) and (6) is simple, so incorporating the trained emulator into existing (or future) analysis codes should be straightforward.

In the limit of a large PCA basis, large training set, and complex neural network architecture, the emulator described above can represent any (deterministic) SPS model to arbitrary precision. However, the power of this emulation framework comes from the fact that, as we will demonstrate in the following sections, a relatively small PCA basis and neural network architecture can achieve percent-level precision over broad parameter ranges, even for relatively complex SPS parameterizations. It is this fact that allows the emulator to achieve such significant speedups.

2.2.3. Discussion

The use of neural networks in this context is solely as a convenient parametric model for an unknown function that we want to learn, in a situation where the dimensionality is too high to make direct interpolation efficient. Neural networks have a number of useful features that make them well suited to this sort of emulation task. The universal approximation theorem tells us that a neural network with a single hidden layer and finite number of nodes can approximate any continuous function on compact subsets of $\mathbb{R}^n$ under some mild assumptions about the activation function (Csáji 2001). Their derivatives can be computed efficiently (by back-propagation), making for efficient training. Once trained, they are straightforward and fast to evaluate, and importantly the computational cost of evaluation is fixed ahead of time and independent of the size of the training set (in contrast to Gaussian processes,[14] where the cost of evaluation naïvely scales as N^3 with the training set size).

In this study, we show that relatively simple dense fully connected network architectures are able to perform well in the context of SPS emulation. However, for more complex SPS models than those considered here, or where fidelity

[14] For use of PCA and Gaussian processes in a similar context, see Czekala et al. (2015).

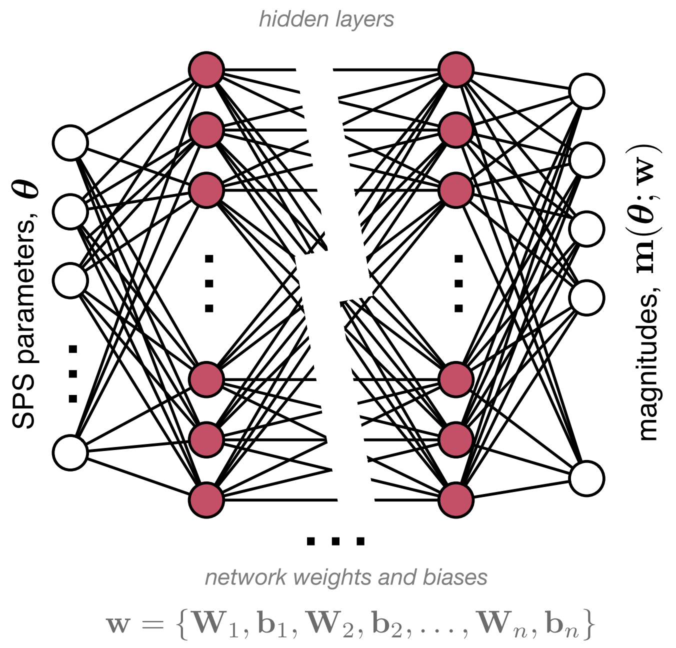
hidden layers
SPS parameters, $\boldsymbol{\theta}$
magnitudes, $\mathbf{m}(\boldsymbol{\theta};\mathbf{w})$
network weights and biases
$\mathbf{w} = \{\mathbf{W}_1, \mathbf{b}_1, \mathbf{W}_2, \mathbf{b}_2, \ldots, \mathbf{W}_n, \mathbf{b}_n\}$

Figure 2. Schematic of the emulator setup for photometry under SPS models; the magnitudes (for some chosen set of bandpass) as a function of the SPS model parameters are parameterized as a dense, fully connected neural network (see Equation (7)).

requirements are very high, more sophisticated architectures may prove beneficial (for a further discussion, see Section 6).

We note that training an emulator on a given SPS parameterization is performed over some predetermined prior ranges for the parameters. Care should be taken to train the emulator over well-chosen priors in the first instance, since emulated SEDs outside of the predetermined prior ranges of the training set should not be expected to be reliable.

2.3. Emulation of Galaxy Photometry

For applications where photometry rather than spectra are the primary target, it makes sense to emulate the photometry directly, i.e., to learn a compact model for the fluxes or magnitudes for some set of filters, as a function of the SPS parameters. Emulating photometry presents a simpler problem than emulating spectra: the number of bands of interest is typically $\mathcal{O}(10)$ (or fewer), so no basis construction or dimensionality reduction is necessary.

To emulate photometry for some set of bandpasses $\{b_1, b_2, \ldots, b_k\}$ under a given SPS model, we parameterize the magnitudes $\boldsymbol{m}(\boldsymbol{\theta}) = (m_{b_1}(\boldsymbol{\theta}), m_{b_2}(\boldsymbol{\theta}), \ldots, m_{b_k}(\boldsymbol{\theta}))$ by a dense fully connected neural network, i.e., (Figure 2):

$$\begin{aligned}\hat{\boldsymbol{m}}(\boldsymbol{\theta}; \boldsymbol{w}) &= a_n(\boldsymbol{W}_n \boldsymbol{y}_{n-1} + \boldsymbol{b}_n),\\ \boldsymbol{y}_{n-1} &= a_{n-1}(\boldsymbol{W}_{n-1} \boldsymbol{y}_{n-2} + \boldsymbol{b}_{n-1})\\ &\vdots\\ \boldsymbol{y}_1 &= a_1(\boldsymbol{W}_1 \boldsymbol{\theta} + \boldsymbol{b}_1),\end{aligned} \tag{7}$$

where $\hat{\boldsymbol{m}}(\boldsymbol{\theta}; \boldsymbol{w})$ denotes the neural network emulated photometry. As before, the weight matrices and bias vectors for each network layer are denoted by $\boldsymbol{W}_k \in \mathbb{R}^{h_k \times h_{k-1}}$ and $\boldsymbol{b}_k \in \mathbb{R}^k$, and we use $\boldsymbol{w} = \{\boldsymbol{W}_k, \boldsymbol{b}_k\}$ as shorthand for the full set of weights and biases of the whole network.

2.4. Activation Function Choice for Neural SPS Emulation

We find that SPS spectra and photometry as functions of the model parameters are mostly smooth but exhibit some non-smooth features. In particular, the behavior as a function of stellar and gas metallicity parameters exhibits discontinuous changes in gradient. When considering neural network architecture choices for SPS emulation, it is therefore advantageous to choose activation functions that are able to capture both smooth features and sharp gradient changes; well-chosen activation functions will allow us to achieve higher fidelity emulation with smaller (faster) network architectures.

To this end, we adopt activation functions of the following form:

$$a(\boldsymbol{x}) = [\boldsymbol{\gamma} + (1 + e^{-\boldsymbol{\beta} \odot \boldsymbol{x}})^{-1}(1 - \boldsymbol{\gamma})] \odot \boldsymbol{x}, \tag{8}$$

where $\boldsymbol{\gamma}$ and $\boldsymbol{\beta}$ are included as additional free parameters of the network to be trained alongside the network weights and biases. This activation function is able to cover smooth features (small β) and sharp changes in gradient (as $\beta \to \infty$). In experiments, we find that activation functions of this form outperform other popular neural network activation choices for the SPS emulation problem (including the tanh, sigmoid, Rectified Linear Unit (ReLU), and leaky-ReLU; see Nwankpa et al. 2018 for recent trends in activation function choice). Nonlinear activation functions of the form Equation (8) are hence adopted throughout this work.

2.5. Target Accuracy for SPS Emulation

While a great deal of progress has been made in reducing modeling uncertainties associated with SPS, some fundamental uncertainties remain (e.g., the effect of binaries and rotation on the ionizing photon production from massive star; Choi et al. 2017; for a review of SPS model uncertainties, see Conroy 2013). When analyzing galaxies under SPS models it is therefore common practice to assume an error floor of 2%–5% on the SEDs or photometry to account for the theoretical SPS model uncertainties (e.g., Leja et al. 2019). On the observational side, for photometry, it is also common practice to put an error floor (typically 5%) on the measured fluxes to account for systematic uncertainties in the photometric calibration (e.g., Muzzin et al. 2013; Chevallard & Charlot 2016; Pacifici et al. 2016; Belli et al. 2019; Carnall et al. 2019).

This context provides a natural accuracy target for SPS emulation (for both spectra and photometry): $\lesssim$5% accuracy or $\ll$5% if we want to ensure the emulator error is a small fraction of the total error budget. While this covers a range of use cases, we note that for an analysis of high signal-to noise ratio (S/N) spectra under very complex SPS models, the fidelity requirements may be more like $\ll$1% (see Section 6 for a discussion).

3. Validation I: DESI Model Spectra

In this section, we demonstrate and validate the emulator on a relatively simple eight parameter SPS parameterization. The model is outlined in Section 3.1, the emulator setup is described in Section 3.2, and validation tests and performance are discussed in Sections 3.3 and 3.4.

3.1. Model and Priors

Our first model (hereafter, the DESI model) is motivated by upcoming analyses of large numbers of optical, low-S/N

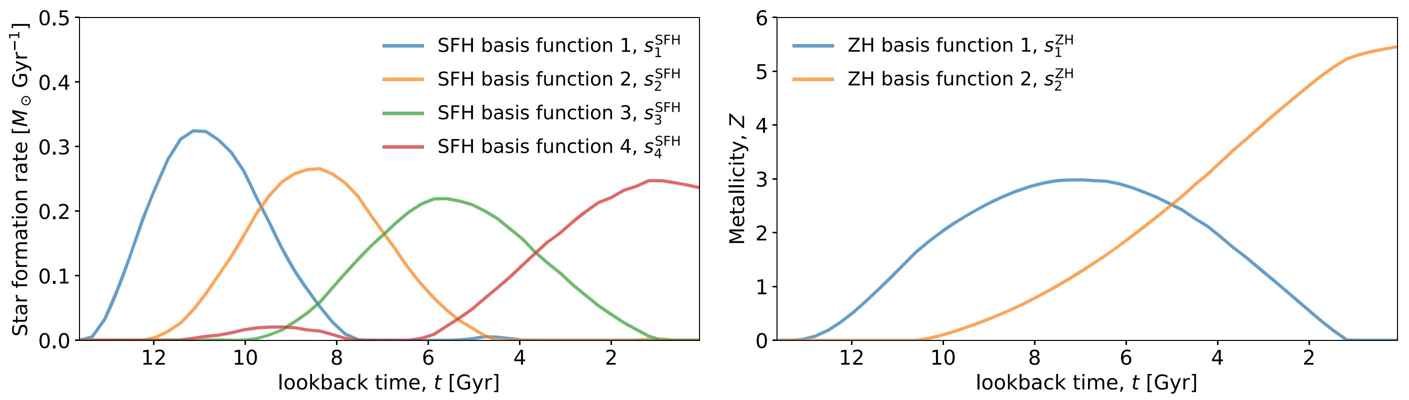

Figure 3. Basis functions for the SFH (left) and metallicity history (right) for the DESI model (see Section 3.1). The SFH basis functions are normalized such that the total mass formed is one solar mass. The metallicity components are unnormalized, but the values refer to the mass fraction in metals ($Z_\odot = 0.019$).

Table 1
Summary of SPS Model Parameters and Their Respective Priors for the DESI Model (Section 3.1)

Parameter	Description	Prior
$\beta_1^{\rm SFH}$, $\beta_2^{\rm SFH}$, $\beta_3^{\rm SFH}$, $\beta_4^{\rm SFH}$	SFH basis function coefficients	Flat-Dirichlet
$\gamma_1^{\rm ZH}$, $\gamma_2^{\rm ZH}$	ZH basis function coefficients	Uniform [6.9×10^{-5}, 7.3×10^{-3}]
$t_{\rm age}$	Age of the universe at lookback time of the galaxy	Uniform [9.5, 13.7] Gyr (equivalent to $0 < z < 0.4$)
$\tau_{\rm ISM}$	Dust optical depth (Calzetti et al. 2000; attenuation model)	Uniform [0, 3]

spectra being collected by current and future surveys. The specifics of the model presented in this section are targeted at the analysis of low-redshift spectra for the upcoming DESI Bright Galaxy Survey (BGS; DESI Collaboration et al. 2016a). The BGS will be a flux-limited survey that will target $\gtrsim$10 million galaxies with $z \lesssim 0.45$ over 14,000 deg^2. It will measure spectra over a wavelength range between 360 and 980 nm with a resolution of $R = \lambda / \Delta\lambda$ between 2000 and 5500, depending on the wavelength. Individual spectra will have a median S/N of ~2–3 per pixel. The key features and free parameters of the model, and associated prior ranges, are as follows.

We model the star formation and chemical enrichment histories as a function of lookback time as linear combinations of a set of precomputed basis functions (Figure 3). The shape and number of basis functions were determined by applying a nonnegative matrix factorization (NMF) to the star formation and chemical enrichment histories of galaxies above 10^9 $M_\odot$ in the Illustris simulation (Vogelsberger et al. 2014a). We sought to construct a basis with the minimal number of components that would reconstruct the history of galaxies, and therefore their optical spectra, to an accuracy dictated by the typical DESI S/N. In practice, the chosen basis has a dependence on the optical colors of the galaxies. The basis used here is an indicative example of what will be used to analyze DESI spectra; further details are given in R. Tojeiro et al. (2020, in preparation).

The star formation history (SFH)[15] for a galaxy at redshift z is implemented as a linear combination of four SFH basis functions, $\{s_i^{\rm SFH}(t)\}$ (shown in Figure 3):

$$\mathrm{SFH}(t;\, t_{\rm age}(z)) = \sum_{i=1}^{4} \beta_i^{\rm SFH} \frac{s_i^{\rm SFH}(t)}{\int_0^{t_{\rm age}(z)} s_i^{\rm SFH}(t)\,dt}, \tag{9}$$

where the SFH basis coefficients $\{\beta_i^{\rm SFH}\}$ are free parameters of the model, the basis functions are normalized to unity over the age of the universe at the lookback time of the galaxy $t_{\rm age}(z)$, and time runs from 0 to $t_{\rm age}(z)$. We train the emulator over a flat-Dirichlet prior for the basis coefficients, i.e., a uniform prior over all combinations of basis coefficients under the constraint that $\sum_{i=1}^{4} \beta_i^{\rm SFH} = 1$ (ensuring that the total SFH is normalized to unity for the emulated spectra).

The metallicity enrichment history (ZH) is similarly parameterized as a linear combination of two basis functions, $\{s_i^{\rm SFH}(t)\}$ (shown in Figure 3):

$$\mathrm{ZH}(t) = \sum_{i=1}^{2} \gamma_i^{\rm ZH}\, s_i^{\rm ZH}(t), \tag{10}$$

where again the ZH basis coefficients $\{\gamma_i^{\rm ZH}\}$ are free parameters of the model, and time runs from 0 to $t_{\rm age}(z)$. We take uniform priors for the ZH basis coefficients, $\gamma_i^{\rm ZH} \in [6.9 \times 10^{-5}, 7.33 \times 10^{-3}]$.

Dust attenuation is modeled using the Calzetti et al. (2000) attenuation curve, with the optical depth $\tau_{\rm ISM}$ as a free parameter with a uniform prior $\tau_{\rm ISM} \in [0, 3]$.

The eight model parameters, their physical meanings, and associated priors are summarized in Table 1.

3.2. Emulation

We generated a training and validation set of 5×10^5 and 10^5 SEDs, respectively, for model parameters drawn from their respective priors (see Table 1) and covering the wavelength range of 200–1000 nm.

The PCA basis was constructed by performing a PCA decomposition of all of the training SEDs.[16] We choose the number of PCA components to keep in the basis such that the

[15] I.e., stellar mass formed per unit time, [$M_\odot$ Gyr^{-1}].

[16] Performing a PCA decomposition over large training sets can be memory intensive. Here we used SCIKIT-LEARN's "incremental PCA," which constructs a PCA basis while only processing a few training samples at a time, keeping the memory requirements under control.

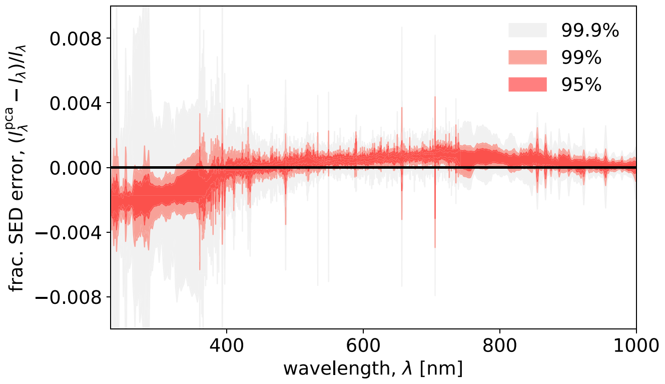

Figure 4. Validation of the PCA basis for the DESI model (Section 3). The central 95% (red), 99% (salmon), and 99.9% (gray) intervals of the fractional errors on the DESI model spectra represented in the basis of the first 20 PCA components are shown. The 20 PCA component basis is able to describe the model spectra to $\lesssim 0.5\%$ accuracy over the whole wavelength range and parameter volume.

basis is able to describe the validation SEDs to $\ll 1\%$ accuracy over the whole wavelength range and parameter volume. Figure 4 shows the fractional error distribution of the validation spectra represented in the PCA basis with 20 components retained; the 20 component basis is able to describe the SEDs to $\lesssim 0.5\%$ accuracy over the whole wavelength and parameter prior range. Note that the PCA basis is constructed for log SEDs, but accuracy in Figure 4 is displayed in linear space.

The PCA basis coefficients are parameterized by a dense neural network with two hidden layers of 256 hidden units, with nonlinear activation functions (Equation (8)) on all except the output layer, which has linear activation. The network is implemented in TENSORFLOW (Abadi et al. 2016) and trained with the stochastic gradient descent optimizer ADAM (Kingma & Ba 2014). Overfitting is mitigated by early stopping.[17]

Network training is performed on a Tesla K80 GPU[18] and takes of the order of a few minutes for the network architecture and training set described above; the computational cost of building the emulator is overwhelmingly dominated by performing the direct SPS computations (using FSPS) to generate the training set ($\sim$10 hr compared to minutes).

3.3. Results and Validation

For validating the trained emulator, we generated 10^5 SEDs for model parameters drawn from the prior and compared the emulated and exact SPS spectra for this validation set. The results are summarized in Figure 5. The upper panels show typical, low, and extreme case performances of the emulator, taken as the 50th, 99th, and 99.9th percentiles of the mean (absolute) fractional error per SED (over the full wavelength range). The bottom left panel shows the 68%, 95%, 99%, and 99.9% intervals of the fractional error as a function of wavelength, and the bottom right panel shows the cumulative distribution of the mean (absolute) fractional error for the validation samples (over the wavelength range). Note that the emulator is trained on the PCA coefficients of log SEDs, but accuracy is shown in Figure 5 in linear space.

The emulator is accurate at the $<1\%$ level over the full wavelength range for $>99\%$ of the SEDs in the validation set. A small fraction ($<1\%$) of validation samples have errors at the few-percent level at the shortest wavelengths. We note that this small number of "outliers" occur where the recent SFH turns on/off and the SEDs are very sensitive to the most recent SFH coefficients. While even in these cases the emulator errors are acceptable, they may be further improved by re-parameterization of the SFH or by better sampling of the prior volume in this part of parameter space.

There are two distinct sources of emulator error: the adequacy of the PCA basis and the accuracy of the neural network in learning the PCA basis coefficients as functions of the SPS parameters. Comparing Figures 4 and 5 (bottom left), we see that the error budget in this case is dominated by the neural network rather than the PCA basis. Accuracy could hence be further improved with a larger neural network architecture (accompanied by a larger training set if necessary), at the cost of some reduction in the performance gain (since a larger network will be more expensive to evaluate).

3.4. Computational Performance

With the network architecture described above (Section 3.2), we find that the trained emulator is able to generate predicted SEDs a factor of 10^4 faster than direct SPS computation with FSPS on the same (CPU) architecture.

Implementation in TENSORFLOW allows the emulator to automatically be called from a GPU, allowing for easy exploitation of GPU-enabled parallelization. Generating 10^6 emulated SEDs takes around $\sim$2 s on a Tesla K80 GPU, compared to $\sim$0.2 s per direct SPS computation on an Intel i7 CPU; an overall effective factor of 10^5 speedup.

When inferring SPS model parameters from galaxy observations, additional performance gains are expected from the use of gradient-based inference and optimization methods that are enabled by the emulator (which has readily available derivatives). We leave investigation of these extra gains to future work.

4. Validation II: Prospector-α Spectra

In this section we demonstrate and validate the spectrum emulator on a more flexible 14 parameter SPS parameterization —the Prospector-α model (Leja et al. 2017, 2018, 2019). The model is outlined in Section 4.1, the emulator setup described in Section 4.2, and validation tests and results are discussed in Sections 4.3 and 4.4.

4.1. Model and Priors

The Prospector-α model includes a nonparametric SFH, a two component dust attenuation model with a flexible attenuation curve, variable stellar and gas phase metallicity, dust emission powered via energy balance, and emission from a dusty active galactic nucleus (AGN) torus. Nebular line and continuum emission is generated using CLOUDY (Ferland et al. 2013) model grids from Byler et al. (2017). Modules for Experiments in Stellar Astrophysics (MESA) Isochrones and Stellar Tracks (MIST) stellar evolution tracks and isochrones

[17] The training set is split 9:1 into training and validation subsets, and the networks are trained by minimizing the loss for the training subset only, but the loss for the validation subset is tracked during training. Overfitting is observed when the validation loss stops improving, while the training loss continues to decrease. Training is terminated when the loss of the validation set ceases to improve over 20 training epochs.

[18] Freely available with Google Colab: https://colab.research.google.com/.

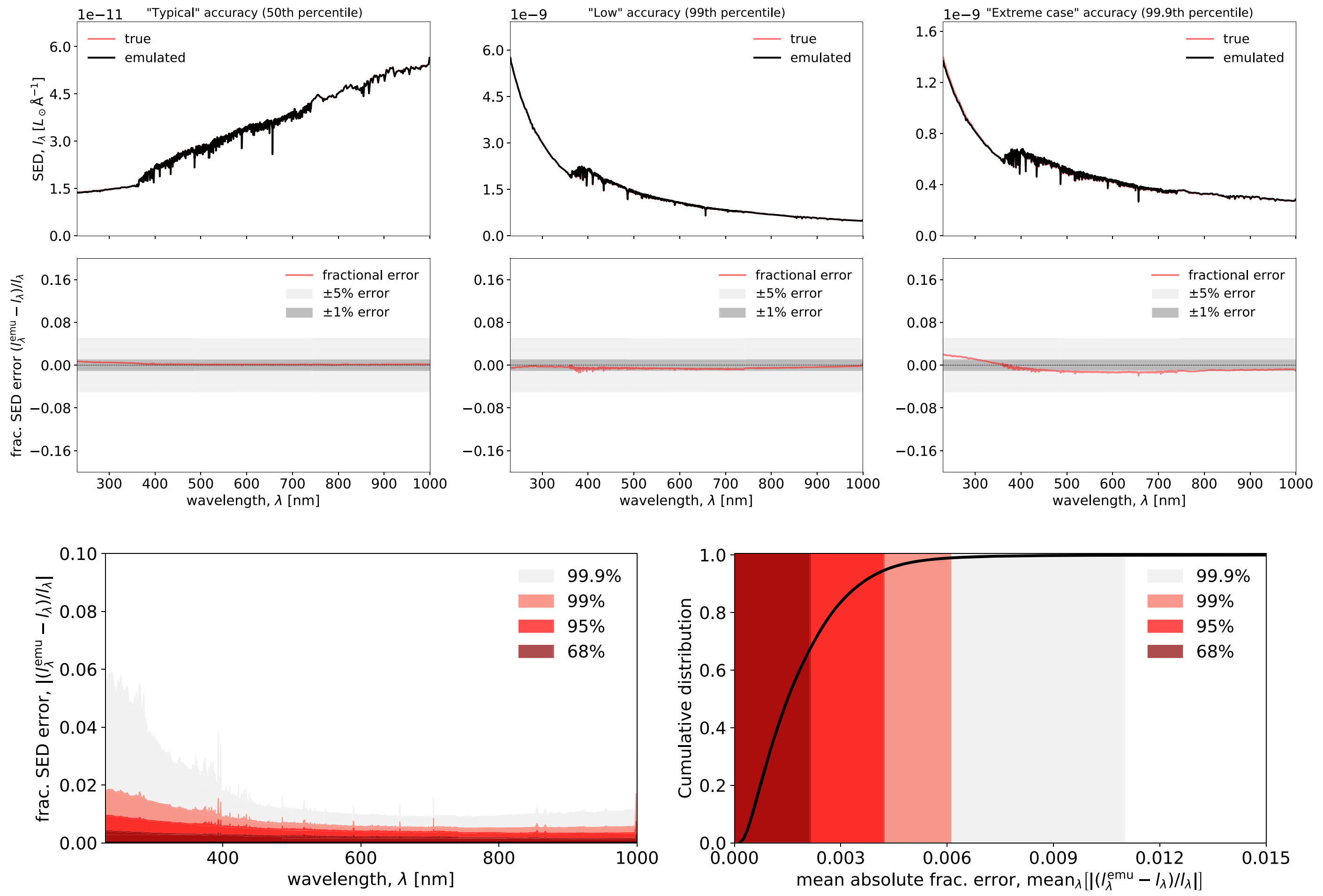

Figure 5. Validation of the emulator for the DESI model (Section 3). Top figure: "typical," "low," and "extreme case" accuracy of the emulated SEDs from a validation set of 10^5 spectra generated with parameters drawn from the prior (Table 1). These cases correspond to the 50th, 99th, and 99.9th percentiles of the mean (absolute) fractional error between emulated and true SED (over the wavelength range). Bottom left: 68 (dark red), 95 (red), 99 (salmon), and 99.9 (gray) percentiles of the fractional emulator error as a function of wavelength. Bottom right: cumulative distribution (blue) and 68 (dark red), 95 (red), 99 (salmon), and 99.9 (gray) percentiles of the mean (absolute) fractional errors (over the wavelength range). We see that the emulator is broadly accurate to $\lesssim$1%, with a small fraction (<1%) of validation samples having errors at the few-percent level or more at the lower end of the wavelength range.

Table 2
Summary of SPS Model Parameters and Their Respective Priors for the Prospector-α Model (Section 4.1)

Parameter	Description	Prior
M	Total stellar mass formed	Log uniform $[10^7, 10^{12.5}]M_\odot$
$r^1_{\rm SFH},\ldots,r^6_{\rm SFH}$	Ratio of log SFR between adjacent bins	Clipped student's-t: $\sigma = 0.3$, $\nu = 2$, $\lvert r^i_{\rm SFH}\rvert \leqslant 5$
$t_{\rm age}$	Age of universe at the lookback time of galaxy	Uniform [2.6, 13.7] Gyr, $(0 < z < 2.5)$
τ_2	Diffuse dust optical depth	Normal $\mu = 0.3$, $\sigma = 1$, min $= 0$, max $= 4$
τ_1	Birth cloud optical depth	Truncated normal in τ_1/τ_2 $\mu = 1$, $\sigma = 0.3$, min $= 0$, max $= 2$
n	Index of Calzetti et al. (2000) dust attn. curve	Uniform $[-1, 0.4]$
$\ln(Z_{\rm gas}/Z_\odot)$	Gas phase metallicity	Uniform $[-2, 0.5]$
$f_{\rm AGN}$	Fraction of bolometric luminosity from AGN	Log uniform $[10^{-5}, 3]$
$\tau_{\rm AGN}$	Optical depth of AGN torus	Log uniform [5, 150]
$\ln(Z/Z_\odot)$	Stellar metallicity	Truncated normal with μ and σ from Gallazzi et al. (2005) mass–metallicity relation (see Section 4), limits min $= -1.98$, max $= 0.19$
z	Redshift	Uniform [0.5, 2.5]

Note. Note that for emulating spectra under this model (Section 4), generated training spectra are computed in the rest frame (but over a range of values for $t_{\rm age}$) and renormalized such that they correspond to $M = 1M_\odot$ (see Section 2 for motivation). When emulating photometry under this model (Section 5), M and z are kept as free parameters to be emulated over.

are assumed (Choi et al. 2016; Dotter 2016), based on MESA (Paxton et al. 2010, 2013, 2015).

The model has been tested and calibrated on local galaxies (Leja et al. 2017, 2018), and recently used to analyze a sample of ∼60,000 galaxies from the 3D-Hubble Space Telescope (HST) photometric catalog (Skelton et al. 2014) over $0.5 < z < 2.5$ (Leja et al. 2019). The model is described in detail in Leja et al. (2017, 2018, 2019); we review the salient features, model parameters, and associated priors below. A summary of model parameters and priors is given in Table 2.

The SFH is modeled as a piece-wise constant, with seven time bins spaced as follows. Two bins are fixed at [0, 30] Myr and [30, 100] Myr to capture recent SFH. A third bin is placed at the other end at $[0.85, 1]\,t_{\mathrm{age}}$, where t_{age} is the age of the universe at the lookback time of the galaxy, to model the oldest star formation. The remaining four bins are spaced equally in logarithmic time between 100 Myr and $0.85\,t_{\mathrm{age}}$. The six ratios of the logarithmic star formation rate (SFR) in adjacent SFH bins $\{r^i_{\mathrm{SFH}}\}$ are included as free model parameters. Following Leja et al. (2017, 2018, 2019) we take independent Student's-t priors on the log SFR ratios (see Table 2). This prior is chosen to allow similar transitions in the SFR as seen in the Illustris hydrodynamical simulations (Torrey et al. 2014; Vogelsberger et al. 2014b, 2014c; Diemer et al. 2017), although care is taken to ensure a wider range of models is allowed than is seen in those simulations.

A single stellar metallicity is assumed for all stars in a galaxy. The observed stellar mass–stellar metallicity relationship from $z = 0$ Sloan Digital Sky Survey (SDSS) data (Gallazzi et al. 2005) is used to motivate the metallicity prior. For a given stellar mass, the stellar metallicity prior is taken to be a truncated normal with limits[19] of $1.98 < \log(Z/Z\odot) < 0.19$, a mean set to the Gallazzi et al. (2005) $z = 0$ relationship, and a standard deviation taken to be twice the observed scatter about the $z = 0$ relationship (to allow for potential redshift evolution in the mass–metallicity relation).

As discussed in Section 2 we fix the integral normalization of the SFH to $1\,M_\odot$ for the spectra in the training set, and stellar mass can then be set by adjusting the normalization of the emulated spectra. However, because in this case the metallicity prior is taken to be mass-dependent, we sample the total stellar mass formed from a log uniform prior from $10^7 M_\odot$ to $10^{12.5} M_\odot$ first (for the purpose of sampling from the metallitiçy prior correctly) and then renormalize the spectra to $1\,M_\odot$ afterward when training the emulator.

Gas phase metallicity is decoupled from the stellar metallicity and is allowed to vary (uniformly) between $2 < \log(Z_{\mathrm{gas}}/Z_\odot) < 0.5$.

Dust is modeled with two components—birth cloud and diffuse dust screens—following Charlot & Fall (2000; see Leja et al. 2017 for details). The birth cloud (τ_1) and diffuse (τ_2) optical depths are free model parameters, with truncated normal priors: $\tau_2 \sim \mathcal{N}(0.3, 1)$ with limits $\tau_2 \in [0,4]$, and $\tau_1/\tau_2 \sim \mathcal{N}(1, 0.3)$ with limits $\tau_1/\tau_2 \in [0,2]$. The power-law index of the Calzetti et al. (2000) attenuation curve for the diffuse component is also included as a free model parameter, with a uniform prior $n \in [-1, 0.4]$.

AGN activity is modeled as described in Leja et al. (2018), with the fraction of the bolometric luminosity from the AGN f_{AGN} and optical depth of the AGN torus τ_{AGN} as free parameters with log uniform priors $\ln f_{\mathrm{AGN}} \in [\ln(10^{-5}), \ln(3)]$ and $\ln \tau_{\mathrm{AGN}} \in [\ln(5), \ln(150)]$, respectively.

The model parameters, their physical meanings, and associated priors are summarized in Table 2.

4.2. Emulation

We generated a training and validation set of 2×10^6 and 10^5 SEDs, respectively,[20] for model parameters drawn from the prior (see Table 2) and covering the wavelength range of 100 nm to 30 μm (using the SPS code FSPS).

To emulate higher-dimensional SPS models over very broad wavelength ranges, such as this case, it is advantageous to split the emulation task into a number of wavelength subranges, which can be stitched together afterward. Here, we will emulate 100–400 nm (ultraviolet (UV)), 400–1100 nm (optical–near-infrared (NIR)), and 1100 nm–30 μm (IR) separately. We find in experiments that that do not split into wavelength subranges, more PCA components are required (in total) to achieve the same consistent accuracy across the full wavelength range. Furthermore, from the perspective of training the neural networks, emulating relatively smaller PCA bases (for each wavelength subrange) represents an easier learning task compared to emulating a single large (>100 component) basis. This means that relatively smaller networks can be used for each subrange, requiring less training data and being faster at evaluating once trained. We do not find any evidence for discontinuities in the emulated spectra at the boundaries between wavelength regions (within the accuracy of the emulator at the boundaries, Figure 7).

The PCA basis was constructed as before by performing a PCA decomposition of all of the training SEDs (for the three wavelength ranges separately), and the number of PCA components retained was chosen such that the resulting basis is able to capture the (validation) SEDs with $\lesssim$1% level accuracy. Figure 6 shows the distribution of errors on the validation SEDs for the PCA basis with 50 components for UV and 30 components for optical–NIR and IR, respectively. This basis is sufficient to describe the SEDs to $\lesssim$1% over the full wavelength range and parameter volume. The errors can be reduced further by increasing the size of the PCA basis but are sufficient for our current purposes. Note that the PCA basis was constructed for log SEDs, but accuracy is shown in Figure 6 in linear space.

The basis coefficients for each wavelength range are parameterized by a dense neural network with three hidden layers of 256 hidden units, with nonlinear activation functions (Equation (8)) on all hidden layers and linear activation on the output. Network implementation and training follows exactly as described in Section 3.2.

4.3. Results and Validation

Similar to the DESI model, for validating the trained emulator, we generated 10^5 SEDs for model parameters drawn from the prior, and compared the emulated and exact SPS spectra for this validation set. The results are summarized in Figure 7. The upper panels show typical, low, and extreme case

[19] Set by the range of the MIST stellar evolution tracks.

[20] We used a larger training set for the Prospector-α compared to the DESI model, owing to the larger parameter space. Training set sizes for both models were chosen so that they could be generated in less than days and achieved percent-level accuracy upon validation. For a further discussion on optimization of training set sizes, see Section 6.

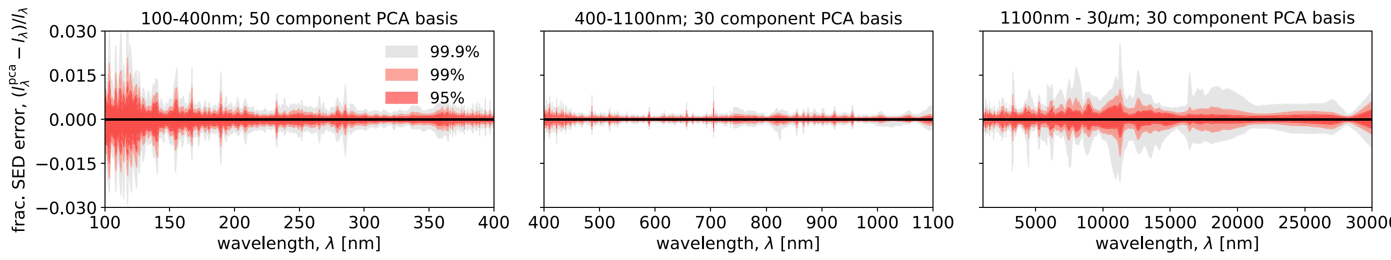

Figure 6. Validation of the PCA basis for the Prospector-α model (Section 4). The central 95% (red), 99% (salmon), and 99.9% (gray) intervals for the fractional errors on the 10^5 validation spectra represented in the basis of the first 50, 30, and 30 PCA components for UV, optical–NIR, and IR wavelength ranges, respectively, are shown. The basis is able to capture the Prospector-α model spectra to $\lesssim 1\%$ accuracy over the entire wavelength and parameter ranges.

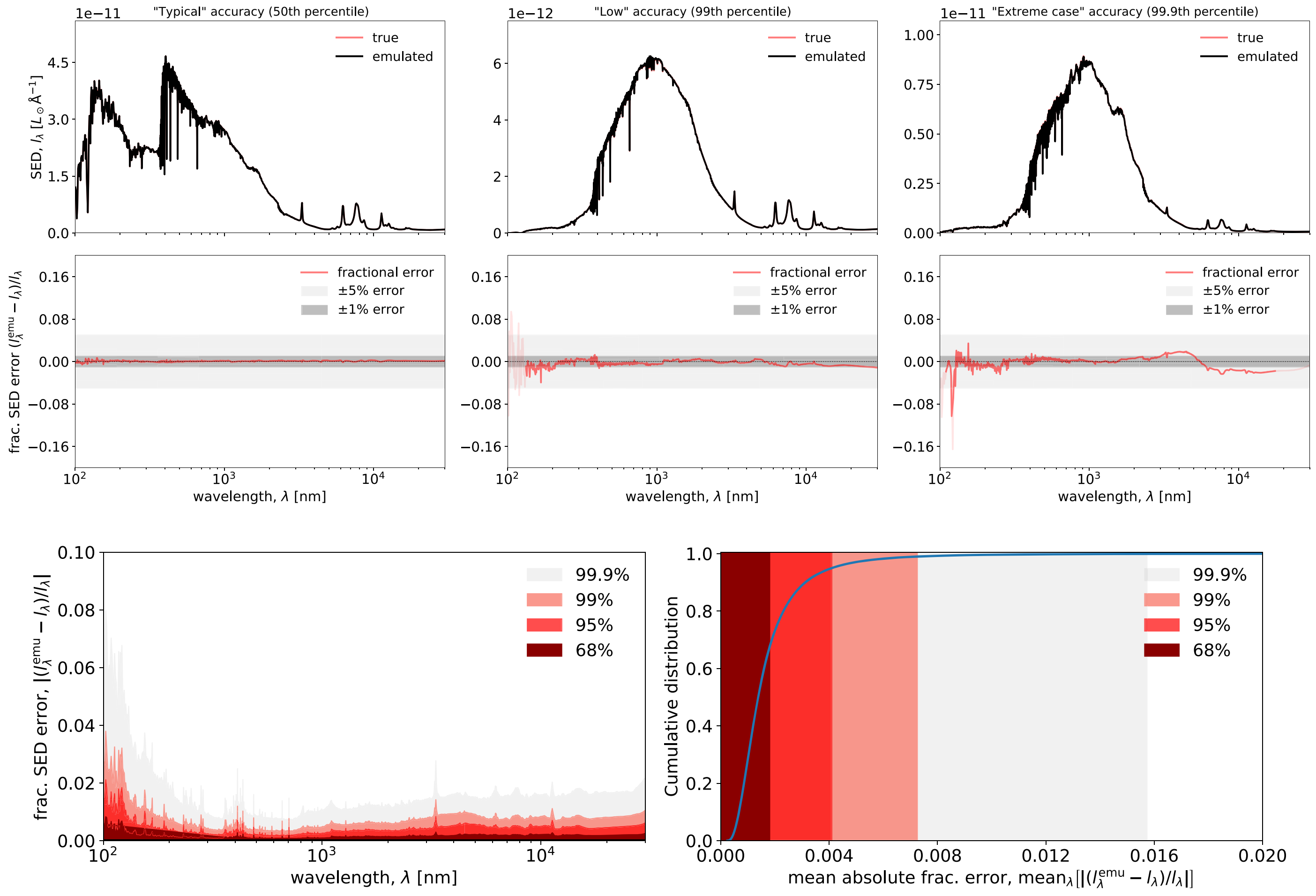

Figure 7. Validation of the emulator for the Prospector-α model (Section 4). Top figure: "typical," "low," and "extreme" case accuracy for the emulated SEDs from a validation set of 10^5 spectra generated with parameters drawn from the prior. These cases correspond to the 50th, 99th, and 99.9th percentiles of the mean (absolute) fractional error between the emulated and true SED (over the wavelength range). The displayed fractional errors (middle row) are faded out where the SEDs $\to 0$. Bottom left: 68 (dark red), 95 (red), 99 (salmon), and 99.9 (gray) percentiles of the fractional emulator error as a function of wavelength. Bottom right: cumulative distribution and 68th (dark red), 95th (red), 99th (salmon), and 99.9th (gray) percentiles of the mean (absolute) fractional errors on the SEDs (over the full wavelength range). Typical errors (68%) are sub-percentages across the whole wavelength range. Of the samples, 99.9% are accurate to <2% over most of the wavelength range, with the tails of the error distribution extending out to ~6% at the shortest wavelengths.

performances of the emulator, taken as the 50th, 99th, and 99.9th percentiles of the mean (absolute) fractional error per SED (over the full wavelength range). The bottom left panel shows the 68%, 95%, 99%, and 99.9% intervals of the fractional error as a function of wavelength, and the bottom right panel shows the cumulative distribution of the mean (absolute) fractional error for the validation samples (over the full wavelength range). Note that the emulator is trained on the PCA coefficients of log SEDs, but accuracy is shown in Figure 7 in linear space.

The emulator has typical fractional SED errors (68th percentile) at the $\ll 1\%$ level over the full wavelength range and parameter volume. Of the validation samples, 99.9% are accurate to better than 2% down to 200 nm, below which the accuracy steadily degrades with tails out to ~6% at the lowest wavelengths (100 nm).

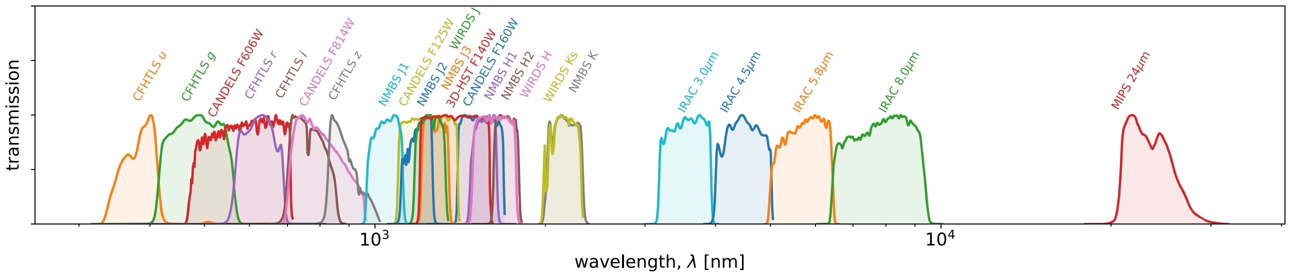

Figure 8. The filters for the 24 bands emulated (for the Prospector-α model) in Section 5, spanning the wavelength range of 300 nm–24 μm.

4.4. Computational Performance

For the Prospector-α model, with the network architecture described in Section 4.2, the emulator is able to generate predicted SEDs a factor of 10^3 faster (per wavelength range) than direct SPS computation with FSPS on the same CPU architecture.

For applications where parallel SPS evaluations can be leveraged, the emulator can be called on a GPU without any additional development overhead. Generating 10^6 emulated SEDs takes around $\sim$2 s on a Tesla K80 GPU, compared to $\sim$0.05 s per FSPS call on an Intel i7 CPU: an overall factor of 10^4 effective speedup per SPS evaluation.

We leave investigation of additional performance gains enabled by the use of gradient-based optimization and inference methods to future work.

5. Validation III: Prospector-α Photometry

In this section we demonstrate and validate direct emulation of photometry on the same Prospector-α model as considered in the previous section (see Section 4.1 and Table 2 for the model and parameters).

For this demonstration, we emulate the 24 bands associated with the All-Wavelength Extended Groth Strip International Survey field for the 3D-HST photometric catalog (Skelton et al. 2014), supplemented by Spitzer/Multiband Imaging Photometer for Spitzer (MIPS) 24 μm fluxes from Whitaker et al. (2014). This is motivated by the recent Leja et al. (2019) analysis of the 3D-HST galaxies using the Prospector-α model. The 24 bands are as follows (shown in Figure 8): Canada-France-Hawaii Telescope Legacy Survey *ugriz* (Erben et al. 2009); Cosmic Assembly Near-infrared Deep Extragalactic Legacy Survey F606W, F814W, F125W, and F160W (Grogin et al. 2011; Koekemoer et al. 2011); NEWFIRM Medium-Band Survey *J1, J2, J3, H1, H2,* and *K* (Whitaker et al. 2011); WIRCam Deep Survey *J, H,* and *Ks* (Bielby et al. 2012); 3D-HST F140W (Brammer et al. 2012); SEDS 3.6 and 4.5 μm (Ashby et al. 2013); Extended Groth Strip 5.8 and 8.0 μm (Barmby et al. 2008); and Spitzer/MIPS 24 μm (Whitaker et al. 2014).

In contrast to spectrum emulation in Section 4 where only rest-frame unit-mass SEDs were emulated (and mass and redshift adjusted afterward as required), when emulating photometry, we keep both mass M and redshift z as free parameters to be emulated over. Recall also that for photometry, we will emulate the apparent magnitudes directly (Section 2.3); there is no need for an intermediate (PCA) basis construction step in this case.

5.1. Emulation

We generated a training and validation set of 2×10^6 and 1×10^5 SEDs and associated photometry, for model parameters drawn from the prior (see Table 2). We parameterized the apparent magnitudes for each band individually by a dense neural network with four hidden layers of 128 units each, with nonlinear activation functions (Equation (8)) on all but the output layer, which has linear activation.

Network implementation and training follows exactly Section 3.2.

5.2. Results and Validation

The performance of the emulator is summarized in Figure 9, which shows the frequency density (black) and 95% (red), 99% (salmon), and 99.9% (gray) intervals of the emulator errors over the validation set for all 24 emulated bands. Across the board, the standard deviations of the error distributions are $<$0.01 mag. For the majority of bands, 99.9% of validation samples are accurate to better than $\lesssim$0.02 mag and better than $\lesssim$0.04 in the worst cases. In applications where an error floor of 0.05 mag is adopted due to SPS modeling and/or photometric calibration systematics, the emulator errors will make up a modest fraction of the total error budget.

5.3. Computational Performance

We find that with the neural network architecture described above, the emulator is able to predict photometry a factor of 2×10^3 faster (per band) than direct SPS computation for the Prospector-α model, with an additional order of magnitude speedup when calling the emulator from the GPU. We find in experiments that larger network architectures give further improvements in accuracy, at the cost of some computational performance, and leave further optimization of network architectures for this problem to future work.

6. Discussion and Conclusions

SPS emulation offers a factor of $\sim 10^3$–10^4 speedup over direct SPS computation, while delivering percent-level accuracy over broad parameter and wavelength ranges. Parallel SPS evaluations can be further leveraged by calling the emulator from a GPU, giving an overall speedup factor of 10^4–10^5 compared to direct SPS evaluations on a CPU (for the models considered). In addition to the direct speedup of SPS calls, the emulated SEDs and photometry come with readily accessible derivatives (with respect to the SPS model parameters), enabling the use of gradient-based inference and optimization methods; this is expected to reduce the number of SPS evaluations required when analyzing galaxy spectra or

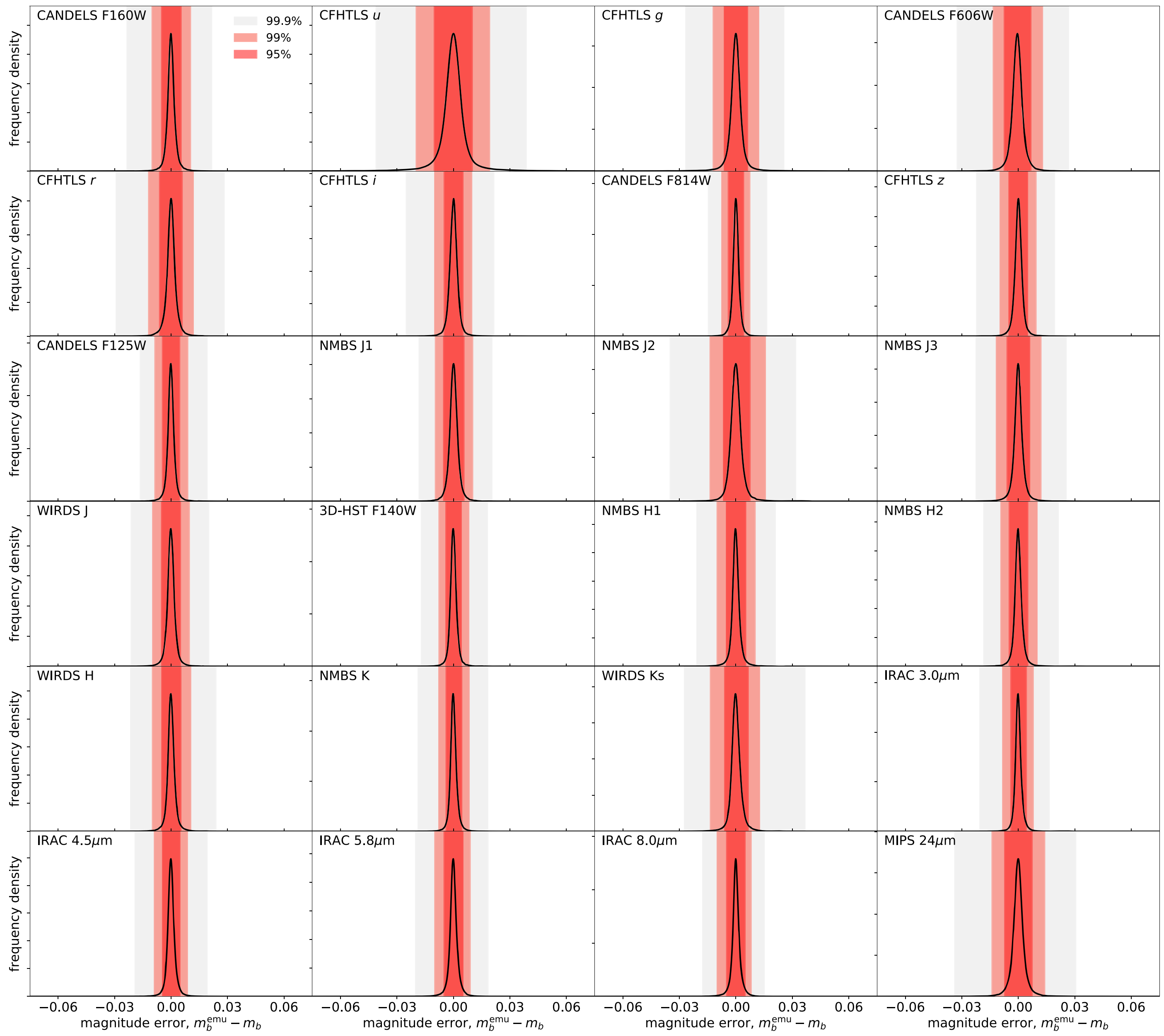

Figure 9. Frequency densities (black) and 95 (red), 99 (salmon), and 99.9 (gray) percent intervals of the errors on the emulated apparent magnitudes for the 24 bands considered (Section 5), over the 10^5 samples in the validation set. For the chosen neural network architecture (Section 5), the emulator is able to deliver percent-level accuracy across the board, with 99.9% of validation samples being accurate to $\lesssim$0.02 mag for most bands and $\lesssim$0.04 in the worst cases.

photometry under SPS models. The implications of the speedup are clear: analyses that previously required significant high-performance computing investment could now be performed on a laptop, and previously intractable analyses of large populations of galaxies will now be tractable. For context, the ∼1.5 million CPU hour analysis of Leja et al. (2019) could now be performed in nearly days on 16 cores, and leveraging the gradients for inference is expected to give additional orders of magnitude improvement on top of that (e.g., Seljak & Yu 2019). Similarly, the computational cost associated with SPS evaluation when forward-simulating large surveys will be radically reduced.

While the specific SPS models presented in this paper were motivated by the analysis of photometry and low-S/N spectra, respectively, another promising area for emulation is SPS models designed to fit high-S/N, high-resolution galaxy spectra. These models are often computationally expensive (∼1 minute per SPS evaluation) and are thus particularly attractive candidates for speedup by emulation. However, the model dimensionality and required precision can be demanding. For the simple case of quiescent galaxies, state-of-the-art models have up to ∼40 parameters that control components such as the initial mass function and individual elemental abundances, as well as detailed models of continuum line spread functions (e.g., Conroy & van Dokkum 2012; Conroy et al. 2018). The systematic residuals for such models are of the order of 1%, so in practice, an emulator would need to reproduce thousands of pixels to sub-percent-level accuracy. Star-forming galaxies bring additional challenges; notably, nebular emission photoionization codes can have hundreds of parameters controlling hundreds of emission lines (Ferland et al. 2017), of which each emission line in principle could have its own line spread function. Although the model complexity and fidelity requirements are higher for this use case, because these models are so much more expensive, one has considerably more leeway in using larger and more sophisticated neural network architectures, while still potentially achieving significant computational speedup.

Another avenue that SPS emulation opens up is a Bayesian hierarchical analysis of large galaxy populations under SPS models, i.e., jointly inferring the physical properties of individual galaxies in a sample along with the intrinsic (prior) distribution of galaxy characteristics. The high-dimensional inference tasks associated with such analyses typically requires gradient-based inference algorithms, such as Hamiltonian Monte Carlo sampling, which will be made substantially easier with emulated SPS.

There are a number of areas where the neural network emulation framework presented here can be improved upon. First, we did not go to great lengths to optimize the neural network architectures to deliver the optimal trade-off between accuracy and speedup. Once the training sets have been generated, training the emulator networks is sufficiently cheap so that a search over network architectures (including more sophisticated architecture types) to deliver the best performance is computationally feasible.

Regarding basis construction for galaxy spectra, we have shown that PCA is effective for a range of applications. However, for complex SPS models or where fidelity requirements are very high, alternative basis constructions such as a NMF in the linear flux (Hurley et al. 2014; Lovell 2019), or nonlinear representation construction with autoencoders, may prove more powerful.

The other area where some additional effort could give substantial improvements is intelligently sampling the parameter space when building the training set. In this study, little focus was given to optimizing parameter space sampling and training set size; training set sizes were simply chosen so that they could be generated in less than days and deliver percent-level accuracy in the trained emulators. However, it is clearly advantageous to use online learning to optimally sample the parameter space on-the-fly in conjunction with the emulator training (see e.g., Alsing et al. 2019; Rogers et al. 2019). This approach has the benefit that it both enables more optimal sampling of the parameter space and, by generating the training set synchronously with training, the size of the training set required to achieve a given accuracy target can be determined on-the-fly (i.e., training and acquisition of training data can be stopped when the accuracy reaches the desired threshold).

For inference applications when the emulator error cannot safely be assumed to be a negligible fraction of the total error budget, it will be desirable to have some quantification of the emulator uncertainties that can be folded into the likelihood function. This can be achieved within the neural network paradigm by using Bayesian neural networks: performing posterior inference over the network weights given the training data (and some priors), hence providing posterior predictive distributions over the output SEDs or photometry rather than simple point estimates. This sophistication comes at the cost of having to perform many forward passes through the network to obtain an emulator error estimate at a given set of SPS parameters.

The emulation code, SPECULATOR, is publicly available at https://github.com/justinalsing/speculator.

We thank Benjamin Joachimi and François Lanusse for useful discussions. J.A. and H.V.P. were partially supported by the research project grant "Fundamental Physics from Cosmological Surveys" funded by the Swedish Research Council (VR) under Dnr 2017-04212. H.V.P., B.L., and D.J.M. acknowledge the hospitality of the Aspen Center for Physics, which is supported by National Science Foundation grant PHY-1607611. J.L. is supported by an NSF Astronomy and Astrophysics Postdoctoral Fellowship under award AST-1701487. This work was also partially supported by a grant from the Simons Foundation and partially enabled by funding from the UCL Cosmoparticle Initiative.

ORCID iDs

Justin Alsing https://orcid.org/0000-0003-4618-3546
Hiranya Peiris https://orcid.org/0000-0002-2519-584X
Joel Leja https://orcid.org/0000-0001-6755-1315
ChangHoon Hahn https://orcid.org/0000-0003-1197-0902
Daniel Mortlock https://orcid.org/0000-0002-0041-3783
Boris Leistedt https://orcid.org/0000-0002-3962-9274
Charlie Conroy https://orcid.org/0000-0002-1590-8551

References

Abadi, M., Agarwal, A., Barham, P., et al. 2016, arXiv:1603.04467
Alsing, J., Charnock, T., Feeney, S., & Wandelt, B. 2019, MNRAS, 488, 4440
Ashby, M., Willner, S., Fazio, G., et al. 2013, ApJ, 769, 80
Barmby, P., Huang, J.-S., Ashby, M., et al. 2008, ApJS, 177, 431
Belli, S., Newman, A. B., & Ellis, R. S. 2019, ApJ, 874, 17
Bielby, R., Hudelot, P., McCracken, H., et al. 2012, A&A, 545, A23
Bishop, C. M. 2006, Pattern Recognition and Machine Learning (Berlin: Springer)
Brammer, G. B., van Dokkum, P. G., Franx, M., et al. 2012, ApJS, 200, 13
Byler, N., Dalcanton, J. J., Conroy, C., & Johnson, B. D. 2017, ApJ, 840, 44
Calzetti, D., Armus, L., Bohlin, R. C., et al. 2000, ApJ, 533, 682
Carnall, A., McLure, R., Dunlop, J., et al. 2019, MNRAS, 490, 417
Charlot, S., & Fall, S. M. 2000, ApJ, 539, 718
Chevallard, J., & Charlot, S. 2016, MNRAS, 462, 1415
Choi, J., Conroy, C., & Byler, N. 2017, ApJ, 838, 159
Choi, J., Dotter, A., Conroy, C., et al. 2016, ApJ, 823, 102
Conroy, C. 2013, ARA&A, 51, 393
Conroy, C., & Gunn, J. E. 2010, ApJ, 712, 833
Conroy, C., Gunn, J. E., & White, M. 2009, ApJ, 699, 486
Conroy, C., & van Dokkum, P. G. 2012, ApJ, 760, 71
Conroy, C., Villaume, A., van Dokkum, P. G., & Lind, K. 2018, ApJ, 854, 139
Csáji, B. C. 2001, in Hyperspectral Image Analysis, ed. S. Prasad & J. Chanussot (Cham: Springer), 48
Czekala, I., Andrews, S. M., Mandel, K. S., Hogg, D. W., & Green, G. M. 2015, ApJ, 812, 128
DESI Collaboration, Aghamousa, A., Aguilar, J., et al. 2016a, arXiv:1611.00036
DESI Collaboration, Aghamousa, A., Aguilar, J., et al. 2016b, arXiv:1611.00037
Diemer, B., Sparre, M., Abramson, L. E., & Torrey, P. 2017, ApJ, 839, 26
Dotter, A. 2016, ApJS, 222, 8
Erben, T., Hildebrandt, H., Lerchster, M., et al. 2009, A&A, 493, 1197
Ferland, G., Porter, R., van Hoof, P., et al. 2013, RMxAA, 49, 137
Ferland, G. J., Chatzikos, M., Guzmán, F., et al. 2017, RMxAA, 53, 385
Foreman-Mackey, D., Sick, J., & Johnson, B. 2014, python-fsps: Python Bindings to FSPS v0. 1.1, Zenodo, doi:10.5281/zenodo.12157
Gallazzi, A., Charlot, S., Brinchmann, J., White, S. D., & Tremonti, C. A. 2005, MNRAS, 362, 41
Grogin, N. A., Kocevski, D. D., Faber, S., et al. 2011, ApJS, 197, 35
Han, Y., & Han, Z. 2014, ApJS, 215, 2
Hurley, P., Oliver, S., Farrah, D., Lebouteiller, V., & Spoon, H. 2014, MNRAS, 437, 241
Kalmbach, J. B., & Connolly, A. J. 2017, AJ, 154, 277
Kingma, D. P., & Ba, J. 2014, arXiv:1412.6980
Koekemoer, A. M., Faber, S., Ferguson, H. C., et al. 2011, ApJS, 197, 36
Leja, J., Johnson, B. D., Conroy, C., et al. 2019, ApJ, 877, 140
Leja, J., Johnson, B. D., Conroy, C., & van Dokkum, P. 2018, ApJ, 854, 62

Leja, J., Johnson, B. D., Conroy, C., van Dokkum, P. G., & Byler, N. 2017, ApJ, 837, 170
Levi, M., Bebek, C., Beers, T., et al. 2013, arXiv:1308.0847
Lovell, C. C. 2019, arXiv:1911.12713
Muzzin, A., Marchesini, D., Stefanon, M., et al. 2013, ApJS, 206, 8
Nwankpa, C., Ijomah, W., Gachagan, A., & Marshall, S. 2018, arXiv:1811.03378
Pacifici, C., Kassin, S. A., Weiner, B. J., et al. 2016, ApJ, 832, 79
Paxton, B., Bildsten, L., Dotter, A., et al. 2010, ApJS, 192, 3
Paxton, B., Cantiello, M., Arras, P., et al. 2013, ApJS, 208, 4
Paxton, B., Marchant, P., Schwab, J., et al. 2015, ApJS, 220, 15
Rogers, K. K., Peiris, H. V., Pontzen, A., et al. 2019, JCAP, 2019, 031
Seljak, U., & Yu, B. 2019, arXiv:1901.04454
Skelton, R. E., Whitaker, K. E., Momcheva, I. G., et al. 2014, ApJS, 214, 24
Torrey, P., Vogelsberger, M., Genel, S., et al. 2014, MNRAS, 438, 1985
Vogelsberger, M., Genel, S., Springel, V., et al. 2014a, Natur, 509, 177
Vogelsberger, M., Genel, S., Springel, V., et al. 2014b, MNRAS, 444, 1518
Vogelsberger, M., Genel, S., Springel, V., et al. 2014c, Natur, 509, 177
Whitaker, K. E., Franx, M., Leja, J., et al. 2014, ApJ, 795, 104
Whitaker, K. E., Labbé, I., van Dokkum, P. G., et al. 2011, ApJ, 735, 86