Multiple Intelligent Agents for Manufacturing Intensification (MIAMI): A Platform for Ranking Clonal Variation in Upstream Bioprocess Development

A thesis submitted to the University College London

for the degree of

Doctor of Philosophy

Qing Xin Tee

2019

Department of Biochemical Engineering

University College London

Declaration

I hereby declare that the work presented in this thesis is solely my own work and that to the best of my knowledge the work is original, except where otherwise indicated by references to other authors.

Qing Xin Tee

29 March 2019

Acknowledgements

I would like to express my gratitude to Dr Darren Nesbeth and Dr Yuhong Zhou for their support and guidance throughout my PhD studies. My grateful thanks extend to all colleagues in the department of Biochemical Engineering for their support throughout my research.

To my loving parents and parent-in-laws, I am so grateful for your unwavering love and encouragement. Special thanks to my husband who supported my decision to pursue this degree.

Abstract

Antibody-based therapeutics are an important class of biotherapeutics for therapeutic applications. With the rising demand and increase in biotherapeutic products on the market, there lies the need for rapid bioprocess development. Clone selection is a critical and time-consuming step in upstream bioprocess development and it is a critical step to execute accurately. A Multiple Intelligent Agents for Manufacturing Intensification (MIAMI) is proposed to process raw data and evaluate clones of three commonly used host cells, Chinese Hamster Ovary (CHO), *Escherichia coli* (*E. coli*), and *Pichia pastoris* (*P. pastoris*).

A search conducted for an IP-free protein sequence yielded the Anti-hepatitis B antibody. The whole antibody sequence was truncated to create a Fab' fragment. Gene designs for three commonly used host cells, CHO, *E. coli*, and *P. pastoris* were created using the IP-free Anti-hepatitis B Fab' fragment.

The development of MIAMI identifies and addresses the necessity of creating a sophisticated code that evaluates clonal ranking based upon data sets. These data sets were collected using the IP-free Anti-hepatitis B gene designs and an existing AV4 gene design.

The AV4 gene design was transformed into *P. pastoris* and repurposed as an inverse methanol detector. In 50mL shake flask culture, green fluorescence protein was detected when cultivating the AV4 strain using glycerol and sorbitol carbon source, while protein transcription was inhibited when using a methanol carbon source. Data collected from cultivating the AV4 strain in 800µL microtiter plates was used to develop the MIAMI software.

The Anti-hepatitis B gene designs were established and characterized in 50mL shake flasks for *E. coli* and *P. pastoris* and a preliminary attempt to establish the gene design CHO. Using the data collected from automated cultivation of 8 different clones of Anti-hepatitis B *E. coli* and *P. pastoris* strains in 800µL microtiter plates scale using the TECAN, a manual ranking of the

clones was performed. Scaling the cultivation up to 200mL DASGIPs microbioreactors, clonal ranking for both strains remained unchanged.

A code was written in python for the processing of raw data. This was demonstrated on the collected HPLC data sets for the Anti-hepatitis B *E. coli* and *P. pastoris* strains, and the flow cytometer data set for the AV4 strain. Multiple agents were created for the development of MIAMI. An assay agent was created for analysing raw HPLC and flow cytometry data to identify and remove unwanted clonal variations. A scanning algorithm calculated the mean and standard deviation of the yields at three consecutive time points to identify a period of stable yield. A ranking algorithm takes into consideration the maximum stable yield achieved and the variability in the data point, giving these two factors a 75% and 25% weighting, MIAMI identifies the best performing clone.

The MIAMI ranking came to the same conclusion as manual human ranking. The effectiveness of MIAMI was validated on the Anti-hepatitis B *E. coli* strain, being able to correctly identify a top performing clone with an optimal induction time, with a conservative estimate of 87% decrease in time taken when compared to manual evaluation.

The MIAMI software significantly improved the timeliness of bioprocess development by accurately screening and evaluating clones. This frees up the time of the user while removing potential sources of human error. With the incorporation of further bioprocesses, MIAMI will become a powerful and effective tool for bioprocess development.

Impact Statement

A Multiple Intelligent Agents for Manufacturing Intensification (MIAMI) is developed to greatly accelerate timeliness of bioprocess development. It utilises a combination of high-throughput methods to execute the screening procedure and intelligent agents to analyse and interpret the results. This effectively reduces the time any researcher needs to perform manual lab work and data analysis for clone selection by approximately 83%.

Development of a bioprocess for any biotherapeutic product is an undertaking that requires considerable investment, not just in time but also in costs. The bioprocess must be optimised to produce the biotherapeutic of an acceptable quality and quantity, while keeping the cost of manufacturing low. Cell line development is the first step of bioprocess development and its relative success would impact efficiency of the rest of the bioprocess. Creating a cell line requires significant consideration as to the viability of the design. Once the design is finalised, clonal variation between clones of the same cell line present a substantial challenge. The difference in behaviour of clones cannot be easily understood and predicted. Thus, optimal clones require a rigorous screening process to identify. This process is laborious and generates a lot of data that requires interpretation.

MIAMI be used to autonomously process through the influx of data effectively and efficiently. The number of clones screened can be increased dramatically without a significant investment in researcher time. Its ability to rapidly and accurately screen a large selection of clones makes it an efficient tool for use in developing commercial manufacturing bioprocesses. In addition to its contribution to the industry, MIAMI also pushes the advancement of software intelligence within the synthetic biological field.

Table of Contents

List of Figures

List of Tables

Abbreviations

BMG	Buffered Minimal Glycerol
BMGY	Buffered Glycerol-Complex Medium
BMS	Buffered Minimal Sorbitol
BMSY	Buffered Sorbitol-Complex Medium
BMMY	Buffered Methanol-Complex Medium
BMY	Buffered Minimal Methanol
CHO	Chinese Hamster Ovary
DO	Dissolved Oxygen
DOE	Design of Experiment
DSP	Downstream Processing
E. coli	*Escherichia coli*
Fab’	Antibody Fragment
GUI	Graphic User Interface
HPLC	High Pressur Liquid Column
HT	High-Throughput
HTS	High-Throughput Screening
IPTG	Isopropyl Beta-D-Thiogalactoside
IRES	Internal Ribosome Entry Site
IVS	Synthetic Intron
LB	Luria-Bertani
mAb	Monoclonal Antibody
MIAMI	Multi-Platform Intelligent Agent for Manufacturing Intensification
OD	Optical Density
OFAT	One Factor at a Time
QbD	Quality by Design
PBS	Phosphate Buffered Saline
P. pastoris	*Pichia pastoris*
USP	Upstream Processing
YPD	Yeast Extract Peptone Dextrose

Chapter 1: Introduction

1.1 Knowledge Based Process Development in Advanced Manufacturing

Process development is a critical step that is imperative for commercial manufacturing of any new product such as food, medicines, electronic devices and automobiles. Production is optimised to ensure the end products are of high quality and are manufactured in a reproducible manner. The development of a new production process is inherently multi-disciplinary. It spans across different individual processes, wherein its successes are measured against different goals and performance requirements. Process development must address factors such as performance, safety, operability, and controllability (Sommerfeld & Strube, 2005).

Prior to process development, quality criteria are set for the product based upon a physical or conceptual prototype and functional parameters are determined for process performance. These parameters are generally based on governmental regulatory requirements, for example Food and Drug Administration guidelines, for product quality and the company's demand for profitability and timeliness. Process development aims to introduce various constraints that will produce the desired objective function that, at the very least, achieve the set minimal target at a manufacturing scale (Jain & Kumar, 2008).

Choices made within process development must coincide with the availability of equipment choice and feasibility of process design. Thus, knowledge-based evaluations of these factors are critical for successful process development. Pisano (1994) suggested there is an empirical link between cumulative production experience and resultant manufacturing performance. Understanding of the product and process naturally lends aid to optimisation efforts towards the predetermined parameters. Adapting the process based upon knowledge-based changes to uncertain elements can reduce wasteful efforts and time delays, achieving a speedier development process.

Considering all possible factors within the process will reduce the amount of uncertainty while maintaining overall performance. An easily replicable process and reproducible results is critical for process development in terms of product profitability, product quality and timeliness (Tatikonka & Montoya-Weiss, 2001). Using a knowledge-based approach, experience can be used to make informed choices to reduce uncertainty within the process.

1.2 Bioprocess Development for Biotherapeutics

There are two main types of biological products, classified as small and large molecules. Small molecules have generally low molecular weight, such as aspirin at 180 Da, and are usually synthesized using traditional organic chemical methods. Biotherapeutics refers to larger molecules with generally high molecular weight, such as monoclonal antibodies at approximately 150 kDa, which are produced from genetically engineered cells. (Chhina, 2013)

Since biological products are intended for human use, each newly discovered biological product must be placed through an additional sequence of tests to determine its safety, efficiency, and proper dosage strength and form. The flowchart shown in Figure 1.2A shows that bioprocess development must occur in parallel to clinical development.

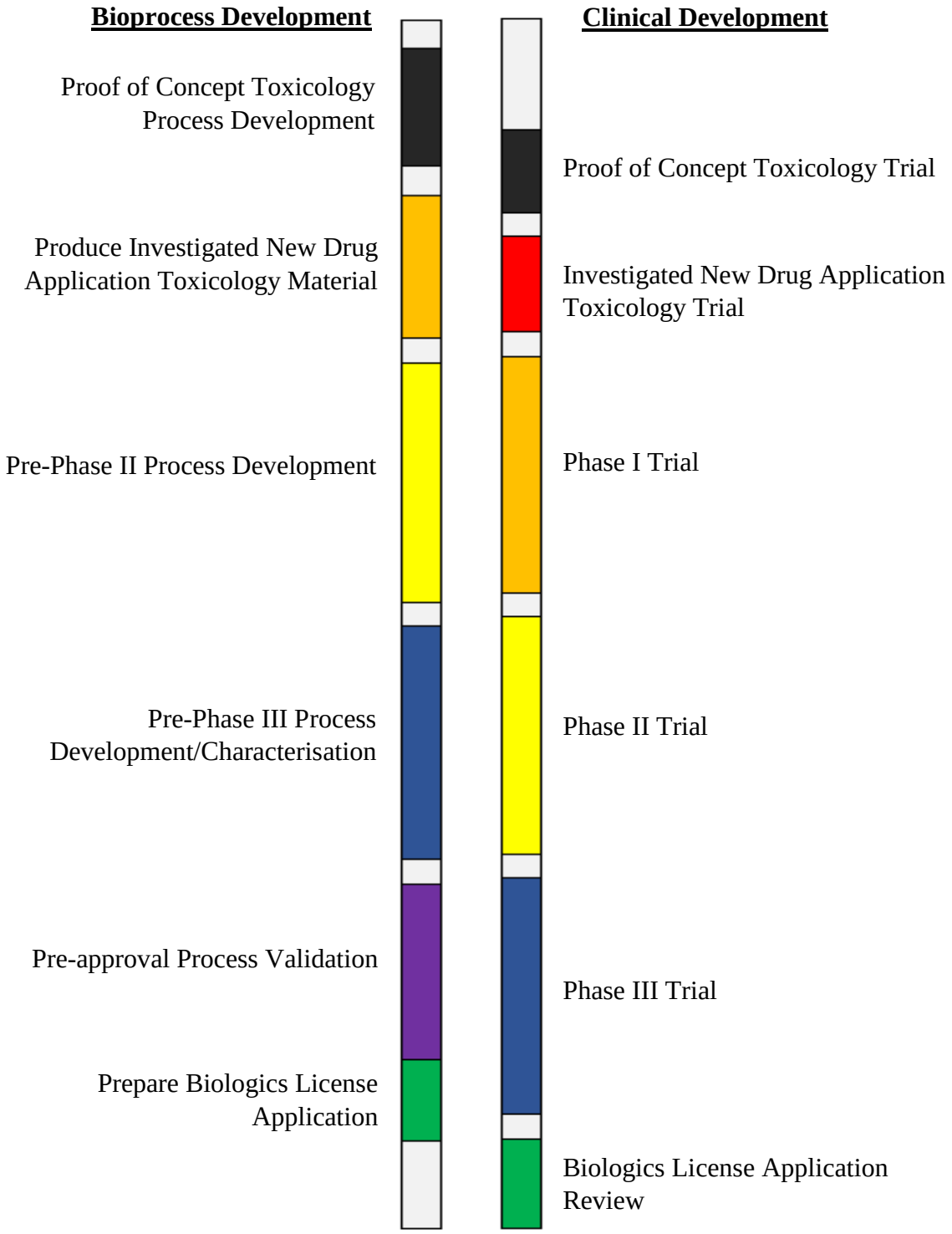


Figure 1.1 Bioprocess and Clinical Development Timeline. Flowchart to depict the parallel nature of process development and clinical development. Bioprocess development starts with proof of concept toxicology process development, followed by the production of toxicology material, pre-phase II process development, pre-phase III process development and characterisation. The finalised bioprocess is then validated and submitted as part of a biologics license application. Clinical development starts with proof of concept and the application of toxicology trials, followed by phase I, phase II and phase III clinical trials. The information gathered in the clinical trials are reviewed and prepared for submission for biologics licence application.

1.3 The Unique Challenges of Bioprocess Development for Biotherapeutics

1.3.1 Aligning Bioprocess Development with Clinical Development

While the biological product can be initially produced in small quantities at a laboratory scale to cope with the volumes required in the early stages of clinical trials, eventually it must be produced in large quantities in extremely pure forms at economically feasible costs and within relevant regulatory constraints. Thus, bioprocess development must occur in parallel with the clinical trials and must scale up in concurrence with each phase. There is an added necessity for continuous manufacture of the biological product for use in clinical trials. This creates an additional challenge while keeping up with the deadlines imposed on bioprocess development.

The additional deadlines are harder to adhere to when dealing with larger molecules such as biotherapeutics. Tertiary structure of larger size of the molecules is critical to their function (Yin, et al., 2007). However, due to the complexity of protein folding, the correct formation of these tertiary structures is challenging to achieve in a consistent manner. Additionally, being synthesized using live cells, biotherapeutics come with an inherent viral contamination risk. These two factors result in more critical process steps than small biological products, effecting the overall timeliness of bioprocess development (Chhina, 2013).

1.3.2 Unpredictable Nature of Cultivation Using Live Cells

A challenge unique to bioprocess development is the use of biologically engineered host cells. The primary aim of bioprocess development is to increase the titre. However, the responses of the cells are unique and cannot be accurately predicted with historical information. This reduces the effectiveness of a knowledge-based approach and thus presents the added challenge unique to bioprocess development. Additionally, changes in the environment or process are less reproducible, presenting a major limitation for scale-up during process development and optimisation.

When dealing with biological organisms, behaviour is heavily contextual and biological processes must display understanding for complex biological networks, accounting for phenomena such as changes in metabolic pathways and expression or repression of genes (Shioya, et al., 1999; Morris & Segal, 2009). While many processes are routine enough to allow for statistical characterisation, the individuality of each biological system makes it challenging to predict its performance using analysis of established similar processes (Adler, et al., 1995).

Bioprocess development possesses a level of complexity and uncertainty that sets it apart from other disciplines (Shioya, et al., 1999). The multifaceted biological networks in bioprocess cannot easily fit into the statistical framework of a traditional knowledge-based approach. This results in a larger difference between processes derived from a knowledge-based approach and the actual optimal process. To bridge this gap, additional efforts would have to be invested to tailor the bioprocess uniquely to its biotherapeutic product. Moreover, operational parameters such as achieving a homogenous temperature and pH are more difficult to control in a biological system (Jain & Kumar, 2008). Consequently, greater investments are required to develop a more sophisticated optimal bioprocess.

1.4 Upstream Processing and Downstream Processing in Bioprocess Development

Process development for biotechnology can be divided into upstream processing (USP) and downstream processing (DSP). USP focuses on optimising for high titres in culture fermentations and efficient harvesting of the bioproduct in primary recovery. In DSP, process development aims to achieve high purification rates with efficient methods to remove by-products and contaminants while guaranteeing product quality. Bioprocess development steps for both USP and DSP aims to decrease investment, time and development cost per antibody. Optimisation steps try to simplify and reduce the number of potential unit operation overall (Jain & Kumar, 2008).

1.4.1 Upstream Processing (USP)

Process development for USP typically starts with the introduction of product encoding transgene, then cell clone selection, media optimisation and lastly bioprocess development and scale up. In recent years, with the increasing advanced options available, reactor choice and process control have also become a critical part of process development (Yang & Liu, 2013). Areas in USP are optimised individually with a focus on robust generation of a high production titre, high productivity and defined quality.

The challenges faced by USP are largely depended on each strains' biological limits rather than equipment capacity limits. Optimisation of USP tries to remove the biological limitations with a focus on achieving higher titres. This is generally achieved by creating or selecting specific cell line and media optimisation. Strains capable of reaching higher titres can be grown in the same equipment and volumes set up. Consequently, higher productivity is achieved with minimal to arguably no impact on the overall processing time and process cost (Gronemeyer, et al., 2014).

1.4.2 Downstream Processing (DSP)

Process development for DSP focuses on the optimisation of individual process for purifying the product, such as virus inactivation, chromatography, and filtrations units. Due to the physical principles for separation, downstream capacity always scales linearly with costs, manifesting itself as the "downstream bottleneck" (Gronemeyer, et al., 2014). Feed volumes in processes in DSP were designed for low titres, and with the significant increase in production titres in recent years leads to DSP equipment reaching its physical limits. This results in an increase in processing time, material consumption and costs. DSP development tries to improve the process capacities of each unit operation while maintaining the yield. The efficiency of the single unit can be achieved by either expanding the existing facility, or through seeking alternative processes (Shukla & Thömmes, 2010).

1.4.3 USP Factors That Influences DSP Performance

Due to the linear nature of process development, optimisation of USP would consequently affect the performance of DSP. As a result, process development for USP would have to anticipate and address its effect on DSP. Producing higher volumes of product to purify in DSP, would directly cause an increase in processing time, material consumption and cost. However, precautionary steps can be taken in USP to lessen the additional burden. For example, developing cell lines with the appropriate post translational modification capabilities to eliminate the need for refolding processes in DSP.

The amount of impurities produced is a significant factor that contributes to the efficiencies of DSP. Host cell proteins constitute a major group of impurities, especially for production of biological drugs. These proteins add a significant workload to DSP, since they must be closely monitored and adequately removed (Jin, et al., 2009). Cell lines can be specifically selected for lower production of host cell proteins. Additionally, changes to medium composition and process conditions, such as temperature and pH, influences the presence of host cell protein impurities. The optimisation efforts in upstream process development would greatly benefit downstream processing time.

1.5 Monoclonal Antibody as an Important Class of Biotherapeutics

Sale of biopharmaceuticals in the United States is expected to have a compound annual growth rate of 11.2% (Yang & Liu, 2013). Monoclonal antibody (mAb) based therapeutics have become an important class of drugs and diagnostic agents due to their specificity and selectivity and are used for many diagnostic and therapeutic applications (Jain & Kumar, 2008; Holliger & Hudson, 2005). They account for a large percentage of the biotherapeutic market, rising from approximately 40% in 2016 to more than 50% in 2018 (Jonakin, 2016; TreDenick, 2018).

The increase in mAbs being regularly approved for therapeutic use, along with the rising market demand, requires a sophisticated but commercially viable process (Jain & Kumar, 2008). Biophysical properties are important parameters that are regulated in the bioprocesses (Kunert & Reinhart, 2016).

Therapeutic mAbs takes the form of a Y-shaped, multidomain protein with the antigen-binding sites located at the tips of the Y, known as the variable region. The variable region binds monospecifically to one antigen, epitope, or cell type, thus determining the function of the mAb (TreDenick, 2018). The therapeutic specificity of the variable region is accountable for the mAb's ability to administer highly targeted therapeutic while minimising side effects.

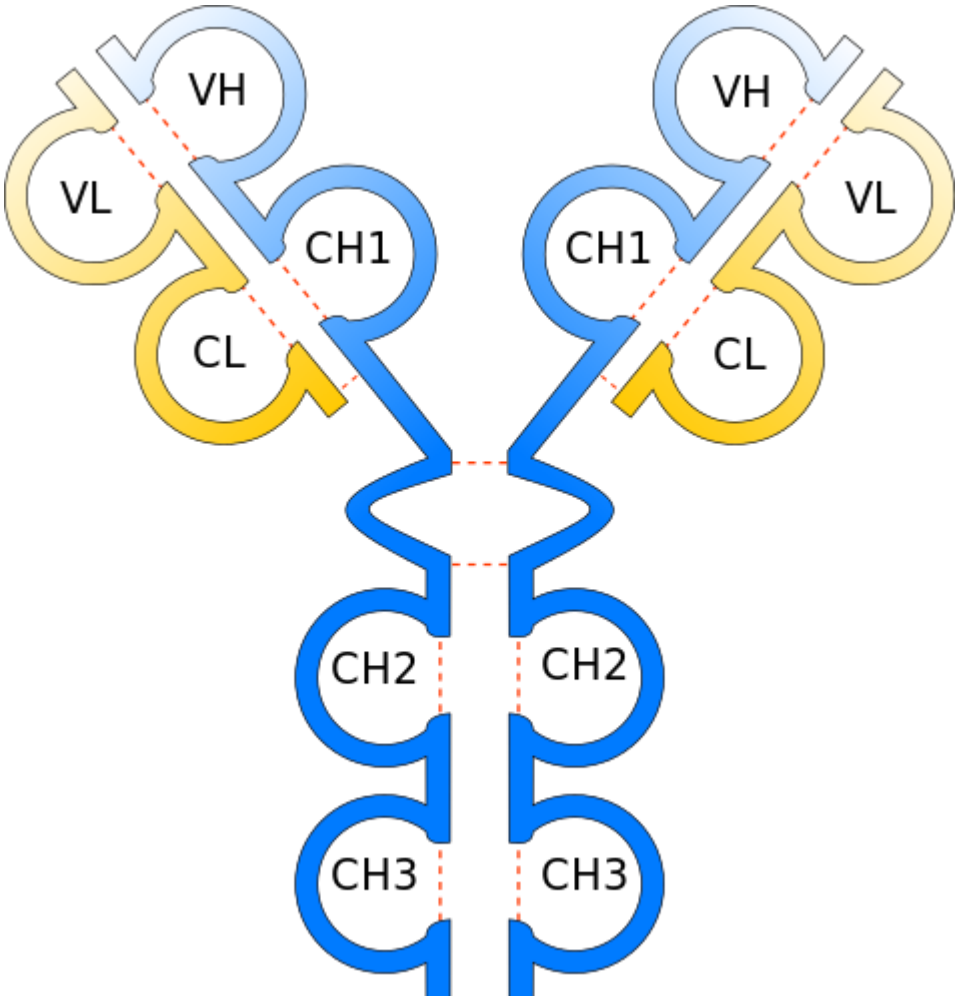


Figure 1.2: Schematic diagram of the basic structure of mAb. The orange represents the light chain, and the blue represents the heavy chain. The dotted red lines are the disulphide bonds. The variable regions of the antibody are labeled as V_H and V_L. The constant regions of the antibody are labeled as C_L, C_H1, C_H2 and C_H3.

1.6 Antibody Fab Fragment (Fab')

An antibody fragments (Fab') contains one constant and one variable domain of each heavy and light chain. They are created by removing the stem region of the antibody cleaving past the disulphide bridge as depicted in Figure 1.6A. This effectively reduces the size of the protein while retaining its therapeutic potency within the variable region.

The smaller size of Fab' allows for the protein to be expressed in more host systems and be more easily secreted. By truncating the constant region of antibody, it removes many glycosylation sites, thus reducing the time required for refolding of proteins. These factors permit smaller hosts cells with less sophisticated folding pathways to be viable alternatives for design of the cell line. When administered, the small size allows Fab' to be more effectively penetrate tissues, making it a more desirable therapeutic choice.

The constant region of the mAb recruits cytotoxic effector, and for most therapeutic applications, these effects are often undesired (Holliger & Hudson, 2005). Thus, not only does the protein not lose any desired qualities from removal of C_H2 and C_H3 region, when administered, undesirable toxic effects are also reduced.

Existing Fab' products includes CDP870 for the treatment of Crohn's disease, rheumatoid arthritis, psoriatic arthritis and ankylosing spondylitis, CEA-Scan for diagnostic imaging of colorectal cancers, and LeukoScan for the soft tissue imaging with a gamma camera (Sieper, et al., 2013; Ghesani, et al., 2003; Ryan, 2002)

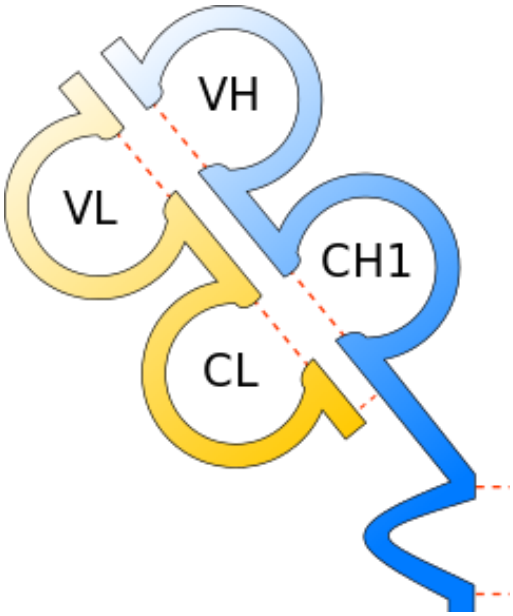


Figure 1.3: Schematic diagram of the basic structure of Fab', created by cleaving at the pepsin on a mAb. The orange represents the light chain, and the blue represents the heavy chain. The dotted red lines are the disulphide bonds. The variable regions of the antibody are labeled as V_H and V_L. The constant regions of the antibody are labeled as C_L and C_H1.

1.7 Bioprocess Development Steps in Upstream Processing for Biotherapeutics

Optimisation for upstream processing expands multiple areas, with a focus on achieving high titres and productivity. Cell line engineering begins with choosing an appropriate host cell. An expression vector is carefully designed for the chosen host cell using synthetic biological techniques. Once a cell line is established, a high performing clone is selected. Culture medium is optimised based upon the clone by identifying the essential nutrients and determining exactly what composition and concentration of each component would be optimal. Process parameters such as temperature, pH and DO are sometimes optimised parallel to medium optimisation.

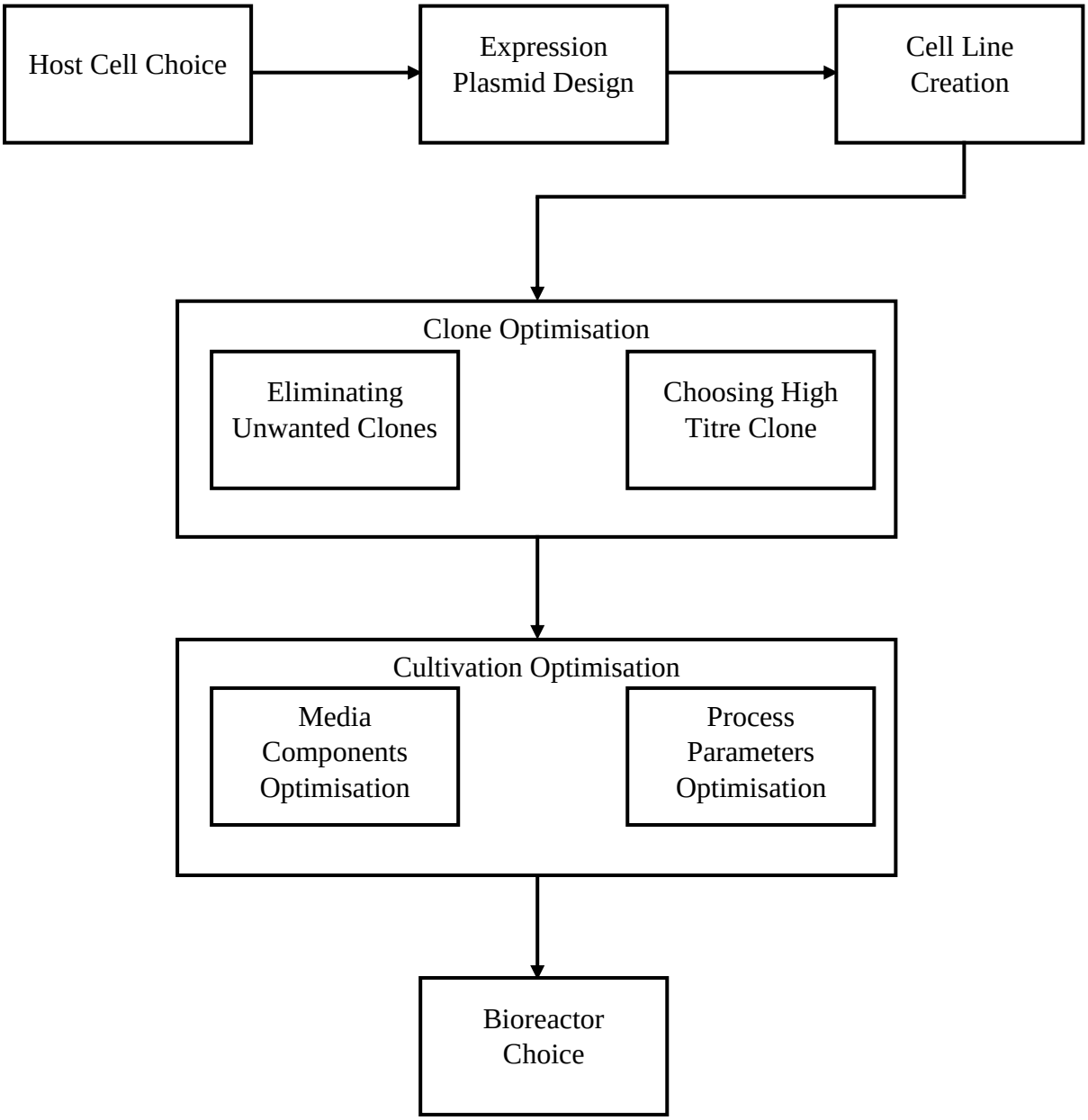


Figure 1.4: Flowchart of Cultivation Optimisation Process. The process begins with cell line development: choosing an appropriate host cell, designing an expression vector and creating the cell line by expressing the protein the chosen host cell. The creation of the cell line is followed by clone optimisation, identifying the unwanted clones and removing them, then choosing a high performing clone. Cultivation parameters such as media components and process parameters are optimised in parallel based upon the chosen clone. Lastly, choosing an appropriate bioreactor for the cultivation.

1.7.1 Major Hosts Cells for Biotherapeutic Cell Line Development

Achievable titre in a process is largely set by the biological limits of the host cell. To address the challenges of removing this limitation, cell line development is a critical part of USP. Selecting the most compatible cell line not only increases the productivity in USP, but also reduces stress on DSP.

Specific production rate can be controlled via genotype and phenotype of the host cell lines (Sommerfeld & Strube, 2005). Genotype is the genetic constitution of the cells whereas the phenotype is the behaviour of the cells as a result of interacting with the environment. The genotype of the cell line is manipulated through the development of DNA design system. The phenotype of the cell would be addressed when developing the cultivation conditions after the cell line is established.

Production of heterologous proteins begins with the development of a suitable cell line. Capabilities of the host system have a major effect on the volumetric productivity (Birch & Racher, 2006). The range of host cells that can be utilised to suit the parameters of any process, coupled with targeted modification of cellular machinery, vastly increased the volumetric and specific productivity for upstream processing (Jain & Kumar, 2008). Consequently, the advancement in protein production has allowed for a multitude of hosts to be viable choices for protein proliferation. There are three commonly used host cells, Chinese Hamster Ovary (CHO), *Escherichia coli* (*E. coli*), and *Pichia pastoris* (*P. pastoris*).

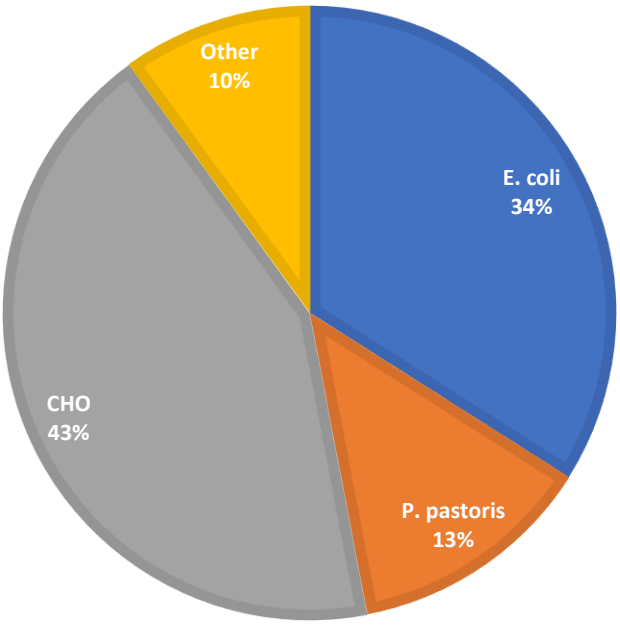


Figure 1.5: Estimation of host cell usage in the European Union and United States biotherapeutics market. With CHO, *E. coli*, and *P. pastoris* being the most commonly used host cells, accounting for 43%, 34% and 13% respectively. The remainder 10% consists of lesser used expression systems such as, *Pseudomonas fluorescens*, *Staphylococcus carnosus*, *Bacillus subtilis*, and *Caulobacter crescentus*. (Meyer & Schmidhalter, 2012)

1.7.1.1 Chinese Hamster Ovary (CHO) Cells

The advantages of mammalian cells as hosts is their ability to correctly perform post-translational modification (Werner, et al., 1998). Mammalian cells have the most complex mechanisms with the ability to promote signal synthesis, secretion and glycosylation of proteins (Yin, et al., 2007). Its larger size also makes them less restricted by the protein size. Thus, is a predominate choice in producing recombinant proteins despite its high expenses and complexity.

Chinese hamster ovary (CHO) cells are most common cells used in the production of biopharmaceuticals. While the proteins might be glycosylated differently due to the difference in species, it is mostly able to yield protein products indistinguishable from naturally occurring ones (Frenzel, et al., 2013). The potential risk of contamination with animal viruses can be eliminated by well-developed Good Manufacturing Practice (Yin, et al., 2007; Frenzel, et al., 2013).

CHO cells are a reliable choice when it comes to the production of heterologous proteins because they have advanced post-translational modification capabilities. This allows them to correctly fold even the most complex proteins. Running costs of CHO cultivations are high due to expensive media components and long doubling time. As a result, high productivity is sought after to reduce the expense per titre. While mammalian expression systems are arguably the most suitable system, a variety of other prokaryotic and eukaryotic hosts are gaining popularity due to their robust nature, rapid growth rate and easier and cheaper culturing conditions (Kunert, et al., 2008).

1.7.1.2 *Escherichia coli*

Prokaryotic expression systems, such as *Escherichia coli* (*E. coli*), have the capacity for continuous cultivation and are very cost effective. As the first host to be approved for pharmaceutical production of recombinant DNA, *E. coli* hosts are preferred due to their low cost and rapid growth (Swartz, 2001). Introduction of foreign DNA into *E. coli* hosts is a straightforward process and require minimal amount of DNA.

However, despite these advantages, *E. coli* lacks the capability to handle complex protein folding. Recent research has sought to characterise and improve upon the folding capability in *E. coli*. As a result, various studies have identified higher yields of correctly folded proteins occurring at lower temperatures and favouring the oxidizing environment of the periplasm (Ukkonen, et al., 2013; Li, et al., 2010; Šiurkus & Neubauer, 2011). This insight led to the deliberate slowing of protein synthesis, lessening host cell strain and promoting the correct formation of proteins (Hsu, et al., 2016). Manipulation of the localisation of the protein encourages secretion into the extracellular medium (Ukkonen, et al., 2013). Both result in higher process yield, improved protein solubility, limited plasmid loss, increased cell viability, and process robustness.

1.7.1.3 *Pichia pastoris*

Pichia pastoris (*P. pastoris*) was recently reassigned as *Komagataella phaffii* (Wei, et al., 2017). However, *P. pastoris* is a more commonly recognised name for this strain. Thus, in this thesis, it will be referred to as *P. pastoris*.

Demand for correctly folded proteins led to the relatively recent host cell alternative, yeast strains. *P. pastoris* is a eukaryotic cell with advanced protein folding pathways for heterologous proteins. As a result, it can produce heterologous proteins in its correct tertiary form, identical

to naturally occurring ones. They also have the advantage of not producing bacterial endotoxins like *E. coli*, and thus satisfy the biosafety regulations for human applications (Yin, et al., 2007).

As both a eukaryotic cell and a microbial, it inherits the advantages of both and serves as an intermediary alternative to CHO and *E. coli* hosts. Its cultivation period, media complexity, and transformation rate are slightly less desirable when compared to *E. coli*, but far more rapid and economical compared to CHO (Lim, et al., 2010; Frenzel, et al., 2013; Yin, et al., 2007). Its rapid growth rate coupled with the relatively small size of individual cells allows for it to achieve comparatively high cell density in bioreactors (Frenzel, et al., 2013).

Although *P. pastoris* is capable of producing proteins identical to naturally occurring ones in terms of binding specificity and affinities, post-translational modifications are not optimal yet. Therefore, not all proteins can be produced successfully in *P. pastoris* (Kunert, et al., 2008). The growth rate, combined with the relatively small size of *P. pastoris* also allows for it to achieve high cell density of up to 0.5 g/L in bioreactors (Frenzel, et al., 2013).

1.7.1.4 Basic Criteria for Host Cell Choice

There are no restrictions on which type of host cells to use. The choice of the best suitable host would depend on the intended product. However, for a strain to be appropriate for commercial manufacturing, it must satisfy quality by design (QbD) criteria. The host cells must be able to achieve a consistent titre with the incorporation of the heterologous protein. Additionally, the host cells must possess the appropriate post-translational processing capability.

1.7.2 General Considerations for Clone Selection

The enhancement of productivity is not only achieved by selecting the most appropriate host, but furthermore by the selection of highly productive clones. While all transformed or transfected host cells should produce the same heterologous protein, the achievable titre and product stability would vary between individual clones (Li, et al., 2010; Sommerfeld & Strube, 2005). While there might be other specific considerations unique to individual strains or product, in general, high productivity and posttranslational processing are the key criteria for cell line selection after the transformation or transfection. The clone chosen for production would ideally produce high titre of correctly folded proteins with consistency.

The general selection process for any strain, *E. coli*, *P. pastoris* or CHO, involves screening of multiple clones, at various scales, not only for metabolic characteristics but also for high stability, robustness, and viability. The bioprocess development must determine which clone to use for production and which would be saved and used as backup (Jain & Kumar, 2008). This can lead to a lengthy process, increasing proportionally with the number of clones screened. The substantial time investment required presents as a significant challenge. Both the selection of the optimum host expression system and clone determines the overall ability of the bioprocess to achieve high productivity and defined quality criteria (Li, et al., 2010). All subsequent process steps are built around the specific clone selected, therefore this is a notably critical decision in bioprocess development.

1.7.3 Medium Development for Cultivation

Generic complex media, which contains a mixture of unknown components, are commercially available for *E. coli*, *P. pastoris* and CHO cells (Gronemeyer, et al., 2014). These complex media, although readily available and easy to use, contains growth chemicals of unknown quantities, leading to many medium components becoming limiting factors for cell growth and productivity (Jordon, et al., 2013). This introduces the need for development towards an

optimised medium for use with a specific cell line, using the generic medium as a basis. The following discussion of medium optimisation significance and methods applies across all three host cell platforms, *E. coli*, *P. pastoris* and CHO, unless otherwise stated.

An optimised medium is subjective, not only to the host strain, but additionally unique to each clone. This exclusivity is due to the high diversity in cell lines, processes, media components, interaction of components and metabolic pathways. For any strain, the optimisation of cell culture medium is an essential step normally performed after clone selection. The improvement from generic medium not only increases productivity of the established bioprocess, but also has an impact on protein quality.

Certain cell medium components have been shown to affect protein structure through reducing glycosylation formation with different strains (Gawlitzek, et al., 2009). Understanding the effects of each component and adjusting its presence in the optimised medium has a great impact on process development. This presents a challenging task as cell culture medium, particularly for mammalian cells, can contain up to 100 different components (Jordon, et al., 2013).

1.7.3.1 Medium Development's Economic Impact

Medium composition plays a critical role in commercialised cultivation processes due not only to its effects on the product, but also on the financial viability of the process (Nor, et al., 2017). Process development is easier in defined medium and it is inherently less expensive while producing fewer contaminants that need to be removed in downstream processing (Baltz, et al., 2010). Thus, the optimisation of medium would make the overall process more economical, a major concern from an industrial standpoint.

1.7.3.2 One Factor at a Time Approach to Medium Optimisation

The traditional approach towards medium optimisation is the simplistic systematic approach known as the one factor at a time (OFAT) strategy. The effects of one component of the medium is characterised while the others are kept constant. This approach can deliver an estimation of the optimum levels but is very time consuming. Since only one factor is observed at a given time, it limits the OFAT strategy's ability to study the interactions between different factors. Interactions between medium components are ubiquitous within complex metabolic pathways, thus requiring additional effort to characterise during medium optimisation (Jordon, et al., 2013). Varying only one factor at any point overlooks these interactions which may lead to inaccurately drawn conclusions, and therefore cannot guarantee the identification of exact optimal conditions (Hu, et al., 2016).

1.7.3.3 Design of Experiments Approach to Medium Optimisation

The alternative method is using factorial designs and in conjunction statistical methods, known as design of experiments (DOE). Several components are varied simultaneously to identify and characterise their interactions and responses. When compared to the OFAT method, DOE can better assess the interrelationship effects between factors and provide a better prediction of the optimal medium, which results in higher growth (Nor, et al., 2017). However, this method requires more experiments to characterise not only the individual factors, but also their interactions with all other factors. The number of experiments required would increase exponentially as the number of monitored factors increases. Considering the workload and risk for human errors, 64 conditions represent an upper limit for manual experimentation (Jordon, et al., 2013).

1.7.3.4 Feeding Strategies for Optimised Medium

While components of the medium can often be optimised at a smaller scale, when scaled up to a bioreactor additional optimisation is carried out regarding modes of feed. Modes of feeding includes batch mode, batch re-feed, fed-batch. Feeding strategy is vital as it provides the cells with the nutrients required for proliferation. Modification in feeding strategy can potentially improve specific productivity, viable cell number and culture periods several folds. Changes in limitations and surplus of medium components effect the type and concentration of impurities. Thus, any modification in medium composition and feeding strategy would also influence the purification process in DSP (Jain & Kumar, 2008).

1.7.4 Bioprocess Parameters of Bioreactors

Therapeutic protein can lose biological activity due to the formation of protein aggregation or the effects of other structural, environmental, and processing factors (Kunert & Reinhart, 2016). Efforts into understanding cell physiology and metabolism have made it possible to identify critical process parameters that affect the productivity of a process. The main parameters that affect cell growths are pH, temperature, and dissolved oxygen (DO). With improvements in online measurement instruments, process control of these critical parameters provides better reproducibility of the biological system. Operational procedures must be developed and controlled to minimise the variability in performance (Glassey, et al., 2000).

Effects of pH and DO appears to be minimal for most cell lines, with trends towards lower cell viability at lower and higher DO (Gomez, et al., 2010; Ozturk & Palsson, 1990; Rosso, et al., 1995). No cell growth occurs above or below specific pH limits. Cell metabolism changes under a certain DO threshold and the decrease in cell viability at low DO correlates with the increase in by-products. It is critical to determine the higher and lower critical limits of the host. Although the change in cell viability and growth does not vary much within the limits, there is

still an optimum point that encourages the highest productivity. Therefore, pH and DO are still important factors to consider for cell viability and growth.

Temperature control is treated in a very similar manner as pH and DO. There is one optimum point operating in a specific range, where beyond the range no cell growth occurs (Rosso, et al., 1995). While the critical range for temperature is still important, recent studies have found that varying the temperature at different points in the bioprocess increases productivity dramatically (Siurkus & Neubauer, 2011). The optimal temperature for cell growth is different to the optimal temperature for protein synthesis; Generally, a traditional higher temperature of 37^{o}C promotes faster cell replication, whereas a lower temperature, as low as 25^{o}C, promotes the production of correctly folded proteins (Ukkonen, et al., 2013; Li, et al., 2014). With the advancements in online process controls, temperature can be varied as required within the process to achieve higher process yield, improved protein solubility, limits plasmids loss, increase cell viability, and better process robustness (Hsu, et al., 2016).

1.7.5 Bioreactors Choices for Cell Culture

With new advancements in terms of bioreactor designs, there is an increase in viable choices for a suitable bioreactor. The selection of a bioreactor considers its ability to provide adequate mass transfer and oxygen supply while imposing low sheer stress on the cells (Sommerfeld & Strube, 2005). Different bioreactors are accordingly suitable for different strains and applications.

A new trend in bioprocessing is the implementation of single-use bioreactor systems. They offer advantages in their lower capital investment and operational costs, alongside higher process replicability. Different forms of the disposable systems include wave bags, orbital shake and stirred tank bioreactors. Although they are limited in their working scale, they eliminate the need for cleaning or sterilisation which significantly reduce contamination rates (Langer & Rader, 2014).

Embracing this technology, several automated scaled down bioreactor systems have been developed. Such as automated work station using ambr^{TM} (Sartorius AG, Göttingen, Germany) systems are small scaled, single use, sterilized, stirred tank bioreactors that are designed to be used in conjunction with an automated workstation. Control measurements can be taken at frequent intervals of 90 seconds, meaning the ambr^{TM} systems are able to hold steady temperature, pH and dissolved oxygen to a high degree of precision and accuracy (Nienow, et al., 2013; Hsu, et al., 2012). The high degree of control and the design of the ambr^{TM} bioreactors have made it possible to reproduce very similar cultivation conditions to large scale bioreactors. Cell growth in these miniature bioreactors outperforms shake flasks by 120%, and trends very similarly to growth in 2 litre bioreactors (Nienow, et al., 2013; Hsu, et al., 2012). As a result, the scalability of the entire cultivation process is very high, while keeping cost of process optimisation low (Rameez, et al., 2014; Amanullah, et al., 2009).

1.8 High-Throughput Approaches for Bioprocess Development

High-throughput (HT) methods became established with the rise of flexibility in process scales, becoming increasingly utilised due to its ability to lessen time pressure in bioprocess development (Bareither, et al., 2013). The very small-scale fermentation, for example using 96-well plates combined with the use of automation equipment, such as TECAN, HPLC and plate readers, allow for parallel runs of multiple experiments at the cost of minimal materials and investment of time. This leads to a large amount of data in a very short period. This method is commonly paired with statistically planned experiments, notably DoE, to optimise the quantity of effective experimental data acquired, or with software that is capable of processing a large amount of data in a fast manner (Sommerfeld & Strube, 2005).

High-throughput screening (HTS) methods are already routinely used for screening multiple cell lines and clones for mammalian expression (Bos, et al., 2015). Individual clones are grown in small volumes in microtiter plates, which allows for parallel growth of multiple clones and as a result, rapid screening. While it is often a time intensive process because of the amount of data generated, it is a necessary tool to accelerate bioprocess development.

Generally, HT methods are used in conjunction with DoE for optimising culture medium components and conditions. Using it to characterize mass transfers, volumetric mass transfer coefficient and oxygen transfer rate to provide valuable information as to how specific productivity varies with different components and medium conditions (Lattermann & Buchs, 2015).

1.9 Implementation of High-Throughput Methods in Quality by Design

There is a rising demand for biotherapeutics for therapeutic applications. To meet this challenge, more efficient process development is required for both upstream cultivation and downstream purification. A critical objective of bioprocess development involves optimising the performance of product manufacturing and implementing a protocol that ensures reproducible product titre and quality.

QbD has gained acceptance within the industry as an approach to developing and commercializing biotechnology (Bhambure, et al., 2011). This involves a well-defined process, established after extensive research, that delivers products with consistent specified quality attributes. Therefore, it is important to deliver a streamlined cell culture platform with standardised process conditions and procedures. Using HT, cell line selection and media composition can be optimised rapidly. This method not only improve outcomes for bioprocess development, but also reduce the risk of failure in the developed bioprocess through means of QbD (Carrier, et al., 2010).

Knowledge-based methods were originally favoured for developing QbD protocols because it reduced the amount of experimentation required to establish an optimum. Working with live cells is less predictable and therefore bioprocess knowledge does not lend itself well to generalisations. With the introduction of HT methods, the number of experiments that can be performed is greatly increased. The increase in the amount of data acquired from screening multiple conditions can be applied effectively and directly to the optimisation of a bioprocess. This brute-forced approach makes the bioprocess development less reliant on contextual bioprocess understanding and shifts bioprocess development away from a purely knowledge-based optimisation.

1.10 Intelligence in Software

Intelligent agents, also known as deliberative agents, have a broad definition that encompasses the simplest to the most complex systems (Meystel & Messina, 2000; Russell & Peter, 2003). The most rudimentary model of an intelligent agent is shown in Figure 1.10A. The information the agent perceives through its sensors and communication from other agents is inputted into the deliberative agent's interface. Within the interface of the agent, there is an imbedded representation of the outer world and a hierarchy of its goals. Using these as basis for its decisions, it is then translated into an action and its output communicated to other agents.

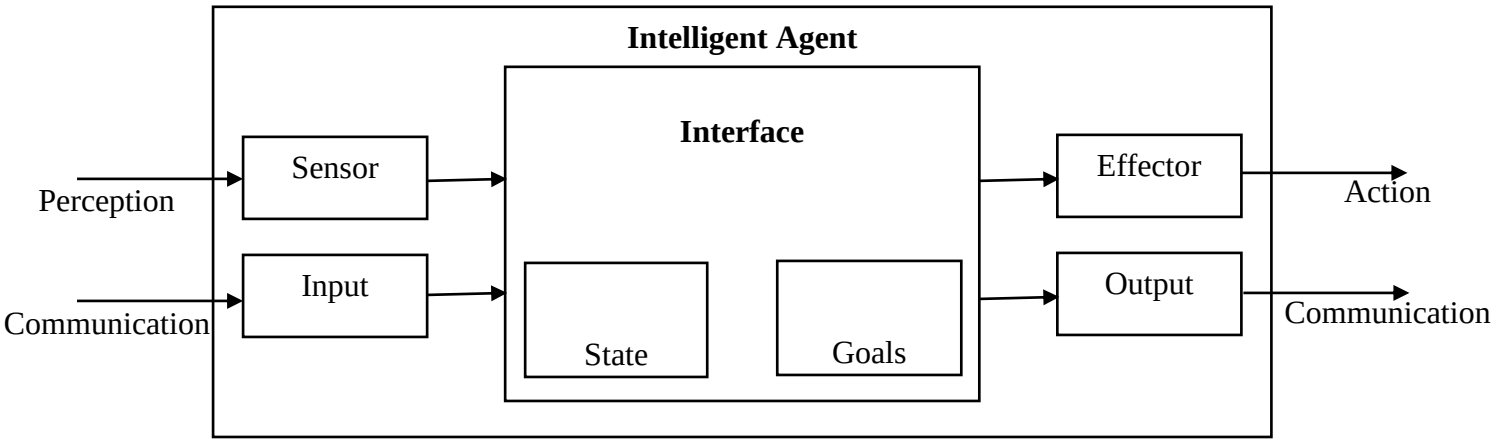


Figure 1.6: Scheme of the architecture of an intelligent agent. The interface contains a stored representation on the state of the outer world and the intended goals of the agent. The agent would process information perceived by its sensors and/or communications from other agents, make a decision based on the state and goals, and materialise these decisions using its effector and/or through communications outputs to other agents.

An example of a simplistic deliberative agent is a temperature control system. In this system, the goal of the agent is to maintain a constant temperature in an area. The representation of the outer world is straightforward, starting and stopping the heater would raise and lower the temperature respectively. If the sensors perceive a decrease in temperature, the agent would decide to start the heating to return the temperature to its set goal. This decision is then actioned by physically starting the heater. The intelligent agent is inherently autonomous. Thus, when the sensor detects that the temperature has returned to the set goal, it would decide to stop the heater.

If there is no temperature sensor built into the temperature control agent, it can still receive input as communication from other agents. A different agent, such as a thermostat, can inform the temperature control agent of temperature changes. The same is true for the output of the decision. If an effector is not imbedded into the deliberative action, it can communicate its decision to another agent, such as an air conditioner. These redundancies might be unnecessary with this simplistic temperature control agent. The flexibility in how the input and output can be received and actioned is crucial when building more complex intelligent agents.

1.11 Current Use of Intelligent Agents to Accelerate Bioprocesses Development

Biological processes have the unique challenge of working with live cells where a high level of variability in the production performance is common and difficult to accurately predict. This is due to the nuanced interaction between the technical choices, cell line selection, operating conditions and capability of the equipment. The success of intelligent agents is determined by the appropriateness of its actions in different situations. Therefore, the bioprocess requires the development of more intelligent methods for practical industrial application.

Bioprocess development requires a deep scientific knowledge and a thorough understanding of process engineering and as a result is prominently driven by experienced personnel creating and evaluating experimental data (Neubauer, et al., 2017). Due to the complexity of biological systems, experimental data generated in the bioprocessing field requires a higher degree of interpretation (Videau, et al., 2010; Skupin & Metzger, 2012). Any resulting adjustments based upon the interpretation would be unique to the biological system used in the bioprocess, thereby, creating a "human in the loop" development process (Glassey, et al., 2000). The interpretation of results will not only be time consuming but would also vary from personnel to personnel, creating a source of variability in the resulting process performance.

An intelligent agent can be developed for more consistent interpretations. However, an elaborate architecture, with communications between multiple agents, is required to compensate for the complex and dynamic nature of a bioprocess system. The redundancies of an intelligent agent, as depicted in Figure 1.10A, are vital in developing an effective software with elaborate architecture comprising of multiple agents. Various agent-based architecture has been developed and validated for different systems would be discussed in more detail in Section 1.12. Multi-agent intelligent architecture allows for different intelligent agents to communicate and works as a collective to collaboratively perform tasks in order to solve a presented complex problem. The modular nature of a multi-agent architecture allows for

complex problems to be divided into more manageable tasks; and since each agent is in theory independent in its design it allows for easier modifications without requiring an overhaul of the architectural software (Gao, et al., 2009; Genesereth & Ketchpel, 1994).

Intelligence within an agent is derived from the translation of the operator's knowledge into codes. Dividing the critical decision points in the bioprocess logically between multiple interacting agents will allow for a more straightforward implementation of codes and algorithms. The agents will represent separate but interacting functions that works to achieve a singular resource target, time target, and bioproduct parameters.

1.12 Intelligent Agents

Advances in automation and HT methods places a bias on raising the experiment throughput rather than addressing the intelligent operations required to interpret the generated data. Systems for bioprocess development must be adaptive and able to make appropriate situational decisions. When discussing intelligent software, the initial thought might be drawn to expecting sensationalised artificial intelligence software with learning capacity. Such as, the robot scientist ADAM with its capacity to develop and test genomic hypotheses, run automated experiments using laboratory robotics, interpret the results to amend the hypothesis and learn from conclusions (King, et al., 2004). While intelligent software like this exist, it is not what intelligent agents refers to.

Intelligent agents, like discussed in Section 1.10, is also known as a deliberative agent. Its core function, as its name would suggest, is to make appropriate deliberations based upon the situation. Its intelligent aspect is reflected through the reasoning and appropriateness of their decisions. This is an improvement in adaptability on traditional software, where decisions are pre-programmed, and the software lacks autonomy over its own reasoning. While intelligent agents can be developed to possess learning capabilities, it is not inherent in its definition.

This section discusses literature that explores the use of intelligent agents in bioprocesses. Creation and implementation of intelligent agents is a novel developing field, as such many of the intelligent agents described are prototypes.

1.12.1 Intelligent Agents for Bioprocess Control

Bioprocess control faces many of the same difficulties as bioprocess development when creating a multi-agent intelligent system. The predominate challenge is creating a model of a bioprocess due to the lack of comprehensive knowledge to fully describe a microorganism and its behaviour (Videau, et al., 2010). Complex interactions between multiple elements and the inherent quasi-optimal environment for cell growth requires an adaptive and dynamic intelligent approach to regulating controls within the bioprocess.

Videau, et al. (2010) described a prototype self-adaptive multi-agent system for bioprocess control. This system comprised of multiple generic cooperative agents. When the agents perceive a change, they receive the information useful to their reasoning, and upon processing their reasoning acts in a useful way towards other agents. This cooperative output reflects the interactions between agents, as such makes the global function of the system adaptive.

Skupin & Metzger (2012) addresses the issues surrounding nonlinear dynamics such as self-sustained oscillation of biomass concentration. The traditional linear approach of detecting the oscillation, determining its source and processing the data before taking an appropriate action to maximise biomass productivity creates a lag period wherein the action taken would not be optimised for the situation by the time it is implemented. An intelligent approach is proposed as a hybrid of three agents and mathematical modelling. Two monitoring agents would detect the oscillation and calculate the average values separately. Using an appropriate mathematical model would pre-emptively determine the action required to reduce the oscillation. The control agent would implement the predictive action that addresses the oscillation in real time, thus providing the bioprocess with more accurate control.

1.12.2 Intelligent Agents for Bioprocess Development

Feng & Song (2002) describes a working intelligent agent prototype for designing a manufacturing process. Although not exclusive to bioproducts, its collective agents use a knowledge base to design a manufacturing process that best satisfy the significant product parameters such as production volume.

Gao, et al. (2009) proposed a theoretical systematic framework for the generation of an agent-based bioprocess development. Using intelligent agents to provide flexibility to overcome the challenges of modelling bioprocess. The intelligent agents have access to the move relevant information and multiple objectives which allows for a rapid bioprocess. A modular set up of this theoretical framework makes it potentially extendable to compass more processes.

Wu & Zhou (2014) described an Intelligent Automation Platform for Bioprocess Development (IAPBD) automated platform consists of four independent but interacting agents: Coordinate Agent, Execution Agent, Experimental Design Agent, and Assay Agent and shown in Figure 1.12.2A. These four agents are bound by the same set of goals and carry out activities to accomplish them.

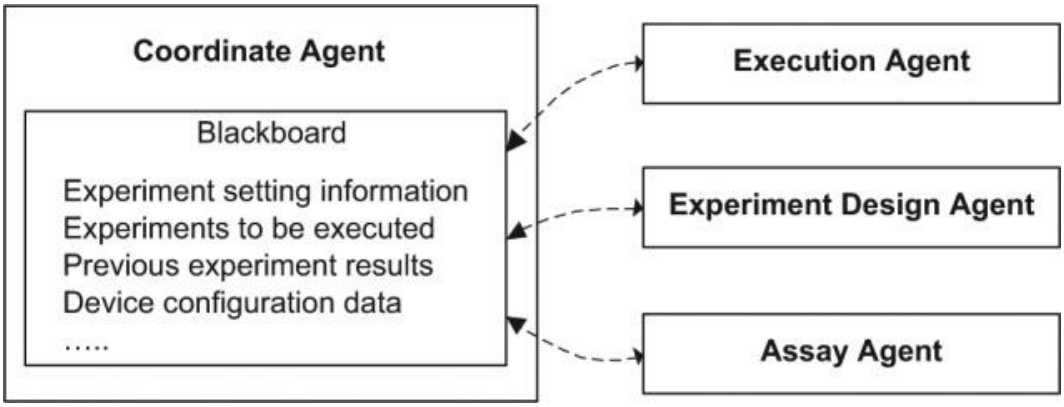


Figure 1.7: Multiagent architecture for the existing intelligent automation platform. The execution agent, experiment design agent and assay agent communicate to each other via the coordinately agent. The coordinate agent will decide tasks based upon the experimental setting information and previous experimental results and assign tasks to the other agents.

IAPBD is a multi-agent based system where the agents have specialised functionalities and work together as a team to deliver a defined bioprocess development task. The communication among the agents was facilitated via Blackboard. The Coordinate Agent is responsible for the publication of information onto a blackboard, which is a hypothetical space where the other agents could withdraw information from and is essential to ensuring the other three agents are working interactively. Before any experimentation begins the experimental parameters would be defined and presented on the blackboard by the Coordinate Agent. Additionally, this agent will search and retrieve historical experimental knowledge from a database based on the predetermined settings and publish the information on the blackboard. This information is subsequently used to direct the other agents. All the agents can publish information on this blackboard, which allows for the other agents to draw upon that information if relevant.

The Execution Agent and Assay Agent are responsible for carrying out the procedures required. The Execution Agent carries out the experimental procedures, for example, create the TECAN scripts to perform pipetting, mixing, moving plates, delivering the plates to plate reader or HPLC etc. The Assay Agent is then responsible for the analytical procedures performed on the samples provided by the Execution Agent. Using devices such as plate reader or HPLC, the Assay Agent would carry out protocols to determine the yield and analysis of raw data. This information would be communicated to the other agents via the blackboard.

The Experimental Design Agent uses the experimental results published on the Blackboard to evaluate the design criteria and decide what to do next. If the objective function is not satisfied, the Experimental Design Agent will design new experiments and publish the new experimental procedures onto the blackboard for the Execution Agent and Assay Agent to execute. If the design criteria are satisfied, the Experimental Design Agent will provide the final solution to the Blackboard. Coordinate Agent will save the data into the database and stop the system.

1.13 Commercial Approaches to Rapid Mapping of Cell and Process Design Space

Various biotechnology companies have incorporated intelligent software into their processes and products with relative success. Most of the software discussed in this section are proprietary.

1.13.1 Ginkgo Bioworks

Ginkgo Bioworks works to translate DNA design into code and store it in a large data base. The data base would contain codes representing DNA components, for example promoters, terminators and secretion signals, and their appropriate order. Using their in-house software, it can draw upon this information to design DNA with different components and combinations, such as varying promotors, secretion signals, terminators, etc. Ginkgo Bioworks is currently utilising this software and TECAN robotics to automate work on organism design across several industries, reporting that they are able to achieve high experimental repeatability with minimal human intervention. They integrated their in-house software to Transcriptic, a software that coordinate scientific processes, instruments and robotics to achieve full lab automation. Using this software, they are able to achieve automated high-throughput strain construction at a small scale.

1.13.2 Silicolife

Silicolife have a focus on developing bioinformatics, which was used to create an artificial intelligence with integrated knowledge of metabolic engineering. Its developed artificial intelligence, with their internally developed automated platform, was able to help generate a novel and proprietary metabolic pathway that is a new cost-competitive alternative for biological production of n-butanol (Lane, 2018). Their work is built upon a multi-objective framework for the optimisation process. Its optimisation algorithms rely on phenotype prediction methods based upon biological assumptions. Their artificial intelligence is able to

select more suitable hosts for specific production targets, leading to high quality products with the added depth of being able to seek trade off solutions for difficult challenges.

1.13.3 Synthace

Synthace have developed the software “Antha”, a biological operating system that can be implemented for multi-factorial DoE optimisation. By controlling existing automated laboratory equipment, such as TECAN, it executes complex workflows that can be reproducible. This software is most notably used within Synthace for DNA assembly. By combining multiple DNA parts directly from clonal input vectors in a one-pot, room temperature reaction, Antha is capable of generating diverse number of clones performed using an automated workflow.

1.13.4 Lab Genius

Lab Genius engineer proteins with enhanced and novel functionality using gene synthesis coupled with machine learning and robotic automation. While new biological designs are created through evolution, there is no guarantee that the mutation will be biologically favourable. Applying intelligence to evolution through machine learning, the EVA software is able to predict which mutations will perform well. The predictive efficiency of EVA improves using novel machine learning algorithms to extract design rules from proprietary and open-access data. Applying this technology, Lab Genius is able to accurately manufacture countless unique DNA sequences without suffering from diversity loss or low fidelity.

1.14 Addressing the Bottlenecks of Clone Selection and Bioprocess Development with MIAMI

Choosing an optimal clone is the initial step of the bioprocess development process and therefore crucial to the overall development timeliness. Changes made to the selected clone would impact product expression and hinder overall process development (Costa, et al., 2010). Thus, selecting the appropriate clone is vital to accelerating bioprocess development. Due to the unpredictable nature of biological screening, high-throughput approach is justified and appropriate for the screening and identification of clones capable of producing high titres (Hubbuch, 2012). Since biological data requires appropriate human interpretation to arrive at a suitable conclusion, HT is a tedious screening method creates significant burden by inundating the researcher with overwhelming information regarding each clone.

A unique challenge for process development in the molecular biology field is the handling of biological products and large quantities of data simultaneously. Overcoming this challenge would lift the rate limiting step in clone evaluation and process development. Multiple Intelligent Agents for Manufacturing Intensification (MIAMI) is a software that use a combination of HT methods and intelligent agent created to address this critical challenge.

MIAMI is intended to be a generic clone selection software for three commonly used host cells in the biotherapeutic industry, as shown in Figure 1.14A. It has the ability to screen multiple clones in parallel using the HT automated equipment. This would provide the information required to evaluate and select for an optimal clone.

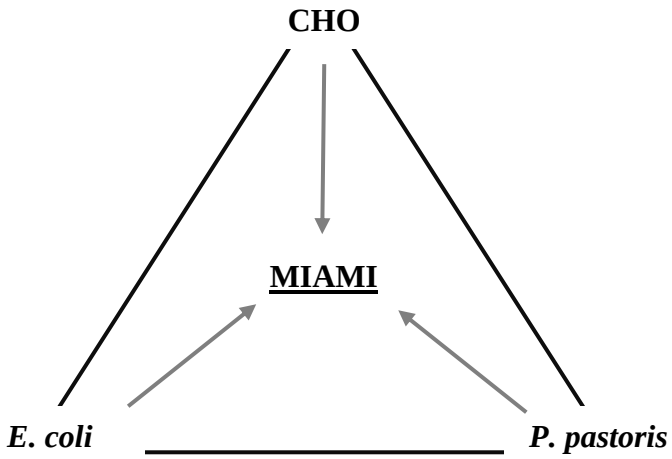


Figure 1.8: Schematic representation of a Multiple Intelligent Agents for Manufacturing Intensification (MIAMI), the bioprocess software intended for this project. This software can receive strains data with CHO, *E. coli*, and *P. pastoris* host cells. It is then able to select for an optimal clone for each host cell.

Clone selection would occur in two stages. In the first instance, a larger number of clones are passed through a rudimentary screening process. In this process, unintended clone variations are identified and removed from selection. The intended clone variations are then ranked by their performance. A reduced sample size of higher performing clones is passed through a more rigorous validation process to generate the data for the evaluation and selection of one optimal clone.

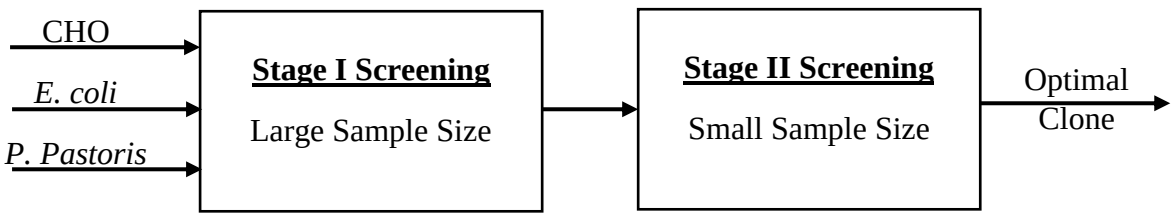


Figure 1.9: Schematic diagram to represent clone ranking process in MIAMI. Data sets from CHO, *E. coli* or *P. pastoris* are passed through Stage I and Stage II screening to generate the selection of an optimal clone.

This project takes an alternative approach to address a different challenge in upstream bioprocess development. Previously discussed in Section 1.3, the challenges faced in upstream bioprocess development is inherently different from those in downstream bioprocess development. The brute-force screening methods allows MIAMI to perceive and identify intended and unintended clonal variations. In doing so, it can select for a clone with the optimal characteristics. Approaching clone selection is such a manner eliminates the need to make assumptions and generalisation of clonal variability, resulting in a truer optimal clone selection process.

The intelligence of the agent would be reflected in its deliberation process for clonal ranking and measured through the appropriateness of its decision. The aim is to achieve an intelligent agent that would interpret the information generated from the brute-force screening in a consistently similar manner to scientific personnel. Data analysis using MIAMI is faster and more accurate than conventional manual screening methods, thus is a great asset to accelerating bioprocess development.

1.15 This Investigation

In this project MIAMI is a software that will be developed as a tool to accelerate bioprocess development in USP. The focus will be placed on accelerating the clone evaluation process in the manufacturing process. The potential to integrate media optimisation and downstream feedback would be considered throughout the project and achievability discussed at the end.

This software would work in tandem with the available automated TECAN platform for cultivate and sample clones, and HT analytical equipment such as HPLC and flow cytometer. The resulting data would be used to make an intelligent assessment of the clones' performance. To support the creation of MIAMI, a typical heterologous protein expression system was constructed for three different hosts: *E. coli*, *P. pastoris*, and CHO. They were used to establish a guideline for the ranking algorithms. A blind comparison would be conducted to establish the appropriateness of MIAMI's ranking output.

1.16 Aims and Objectives of This Thesis

The aim of this thesis is to create the MIAMI software that comprises of multiple intelligent agents that work to select for a high performing clone in three different strain platforms. Each intelligent agent would be created based upon manually collected data. Once the first version of MIAMI is complete, its functionally would be validated using a blind run.

The synthetic gene network AV4 will be repurposed as an inverse methanol sensor, with varying levels of GFP as output. The resulting flow cytometry data is used, in part, to develop MIAMI.

- Validating that the AV4 strain performs appropriately as an inverse methanol detector.

Re-engineering an IP-free whole antibody sequence as a fab' fragment free of glycosylation sites. Genes encoding this fab' are written for expression in CHO cells and will be transfected into suspension CHO cells for characterisation at different scales. Data collected from these cultivations will be used to develop the clonal ranking algorithm in MIAMI.

- Identify an appropriate intellectual property free antibody-based protein sequence.
- Reverse engineering the mAb sequence to create an optimised Fab'.
- Design a Fab' expression system for CHO cells.
- Transfect and cultivate new stable CHO cells expressing the Fab'.
- Evaluate the scalability from 96-well plates to bioreactors for the CHO strain.

Genes encoding the Fab' were written for expression in bacterial cells for transformation into *E. coli* cells for characterisation at different scales. Data collected from these cultivations will be used to develop the clonal ranking algorithm in MIAMI.

- Design a Fab' expression system for *E. coli* cells.
- Transform and cultivate *E. coli* expressing the Fab'.
- Evaluate the scalability from 96-well plates to bioreactors for the *E. coli* strain.

Genes encoding the Fab' were written for expression in yeast cells for transformation into *P. pastoris* cells for characterisation at different scales. Data collected from these cultivations will be used to develop the clonal ranking algorithm in MIAMI.

- Design a Fab' expression system for *P. pastoris* cells.
- Transfect and cultivate *P. pastoris* expressing the Fab'.
- Evaluate the scalability from 96-well plates to bioreactors for the *P. pastoris* strain.

Using the data gathered manually, a new software, MIAMI, wascreated for the ranking of clones.

- Develop code that is capable of processing raw data generated from 96-well plates in multiple formats, such as fluorescence flow cytometry data, High Pressure Liquid Column (HPLC) data and plate reader OD data.
- Develop an intelligent agent that can appropriately assess the performance of different clones and processes.
- Validate the software ranking against human ranking.

Chapter 2: Materials and Methods

2.1 DNA

2.1.1 Microgram Scale DNA Purification Using Miniprep

Miniprep was performed using GeneJET plasmid purification kit provided by Thermo Scientific (Thermo Fisher Scientific, Waltham, Massachusetts, United States). 10mL of culture was prepared for each column used. The culture was incubated in an incubator shaker for 20 hours before it was pelleted at 3300g. The supernatant was decanted carefully, and the remaining pellet was resuspended in 250µL of the provided cell resuspension buffer. The solution was vortexed to ensure a homogeneous mixture then transferred to 1.5mL Eppendorf tubes.

250µL of cell lysis solution was then added and mixed by inverting the tubes slowly but continuously for duration of 2 minutes. Then 350µL of neutralization solution was added and mixed by inverting the solution in a similar fashion for 1 minute. The mixture was then centrifuged at 12500g for 5 minutes. The resulting supernatant was then transferred carefully into the provided spin columns placed in a collection tube.

The supernatant was passed through the columns by centrifuging for duration of 2 minutes at 12500g. The flow through of the column was subsequently decanted, and the columns were placed back into the collection tubes. The spin columns were washed by passing 500µL of wash buffer twice, decanting the flow through from the collection tube each time. After the columns were washed, they were placed in sterile 1.5mL Eppendorf tubes.

50µL of elution buffer, pre-warmed to 50°C, were added to the columns, and left to incubate for 2 minutes at room temperature. The columns were centrifuged at 12500g. The flow through was placed back into the column and was used to elute the column again in the same manner.

2.1.2 Milligram Scale DNA Purification Using Maxiprep

Maxiprep was performed using the plasmid extraction kit provided by Qiagen (Qiagen, Venlo, Netherlands), which contained all the necessary equipment and buffers. A 10mL starter culture prepared in a 50mL falcon tube was left to incubate for 8 hours. This was used to inoculate 250mL of media in a 1L shake flask. The 1L shake flask was left in an incubator shaker overnight.

The overnight culture was then transferred into five 50mL falcon tubes, and spun in a tabletop centrifuge at 6000g for 15 minutes at 4°C. The supernatant was checked visually to be clear and decanted carefully. The remaining pellet was resuspended homogenously in 10mL of the provided Buffer P1. To avoid damaging the plasmid the solution was mixed by pipetting up and down carefully rather than by use of vortex. 10mL of Buffer P2 was added and mixed by inverting slowly but continuously for 2 minutes, then left to incubate at room temperature for 3 minutes. The solution was checked to be of a homogenous blue color.

During this incubation period, the QIAfilter Cartridge was prepared by attaching a cap onto the outlet nozzle and placing upright. After the incubation period, 10mL of chilled Buffer P3 was added and mixed by inverting slowly but continuously for 2 minutes. Once the solution was completely colorless, it was poured into the barrel of the prepared QIAfilter Cartridge. While the solution was left to incubate at room temperature, the HiSpeed Tip was equilibrated by adding 10mL of Buffer QBT to the column and leaving the buffer to allow it to slowly enter the resin.

After the solution had been incubating for 10 minutes, a plunger was inserted to filter the cell lysate into the equilibrated HiSpeed Tip. After the entire volume of lysate had entered the resin, the resin was washed with 60mL of Buffer QC. Lastly, the plasmid was eluted using 15mL of Buffer QF. The eluted solution was precipitated with 10.5mL of isopropanol, this was mixed by inverting the two a couple of times and left to incubate for 5 minutes at room temperature.

The plunger from a 30mL syringe was removed and a provided QIAprecipitator Module was attached onto the outlet nozzle. After incubation, the solution was transferred into the prepared syringe and the mixture was filtered through the QIAprecipitator using the plunger at a constant pressure. The filter was washed by pressing through 2mL of 70% ethanol, and then dried by pressing air through the QIAprecipitator forcefully 5 times. It was vital to ensure that the QIAprecipitator was removed before extracting the plunger each time.

The QIAprecipitator was removed, dried and then attached onto a 5mL syringe. 1mL of Buffer TE, pre-warmed to 50°C, was added and passed through the QIAprecipitator at a slow but constant pressure to elute the DNA into a clean collection tube. The same solution was then passed through the QIAprecipitator another time into the same collection tube, remembering again to remove the QIAprecipitator before extracting the plunger.

2.1.3 Restriction Digest and Electrophoresis of DNA

10X TAE buffer provided by Life Technologies (Life Technologies, Carlsbad, United States) was diluted to 10-fold to create 1X TAE buffer. Ultrapure Agarose, DNA ladder, and DNA loading buffer were all provided by Invitrogen (Invitrogen, Carlsbad, California, United States). All restriction enzymes and associated enzyme buffers were acquired from New England Biolabs (New England Biolabs, Ipswich, Massachusetts, United States).

10µl of purified DNA, 7µl of milliQ water, 2µl of enzyme buffer and 1µl of restriction enzyme were combined in an eppendorf to a total volume of 20µl. The eppendorf tubes were incubated at 37°C for a minimum of 6 hours and maximum of 24 hours.

Prior to removing the DNA mixture from incubation, 0.7g of agarose was dissolved in 70mL of 1X TAE buffer. This solution was cooled and 14µl of 500µg/mL ethidium bromide stock solution was added. The solution was mixed thoroughly and then poured into a cast containing a comb for well formation gently to avoid air bubbles. The gel was left to set for 20 minutes. After the gel solidified, the comb was removed, and an excess of 1X TAE buffer was added to fill the wells and cover the gel.

10µl of DNA ladder was added to the first and last wells. Incubated DNA was combined with loading buffer at a ratio of 4:1 and mixed thoroughly. The resulting solution was loaded into empty wells between the DNA ladder. The gel was set to run for 30 minutes at 150V. Once finished, the gel was transferred to a UV illuminator system to visualize the bands using the ladder as reference.

2.2 *E. coli*

2.2.1 Antibiotics Stock Solutions for *E. coli* Strains

Antibiotic stock solution was prepared in 1000X the recommended concentration and was added to culture media in a 1:1000 dilution. All chemicals products were provided by Sigma-Aldrich (Sigma-Aldrich, St. Louis, Missouri, United States). Stock solutions of 50mg/mL tetracycline, 50mg/mL chloramphenicol, 100mg/mL ampicillin, were prepared, sterilised by passing through a 0.22µm filter and stored in aliquots of 150 mL in -20 °C.

2.2.2 Shake Flasks Cultures for *E. coli* Strains

250mL shake flasks were sterilised in an autoclave. 50mL of sterilised culture and the appropriated antibiotic were added to the flasks. The media was inoculated with the intended strain and placed in incubator shakers for cultivation for approximately 1-2 days. Any handling of the flasks and media during sampling was handled in a sterilised environment inside a biological safety cabinet.

2.2.3 *E. coli* Growth Media

The chemical components of all media were provided by Sigma-Aldrich. Luria-Bertani (LB) broth was composed of 10mg/L tryptone, 5mg/L yeast extract and 10mg/L sodium chloride. Low salt LB was made by reducing the concentration of sodium chloride to 5mg.

2XPY medium was composed of 16g/L phytone, 10g/L yeast extract and 5 g/L sodium chloride.

100X SM6E Trace Elements was composed of 104g/L citric acid, 5.22g/L calcium chloride dihydrate, 2.06g/L zinc sulfate heptahydrate, 2.72g/L manganese(II) sulfate tetrahydrate, 0.81g/L copper(II) sulfate pentahydrate, 0.42g/L cobalt(II) sulfate heptahydrate, 10.06g/L iron(III) chloride hexahydrate, 0.03g/L boric acid, and 0.02g/L sodium molybdate dihydrate.

$SM6G_c$ Medium was composed of 5.2g/L ammonium sulfate, 4.4g/L sodium phosphate monobasic, 4.03g/L potassium chloride, 1.04g/L magnesium sulfate heptahydrate, 10mL/L SM6 elements, 4.16 g/L citric acid, 112 g/L glycerol, 0.25 g/L calcium chloride dihydrate, and 3.35 g/L sodium phosphate. pH of the media was adjusted to 6.95 using ammonia.

2.2.4 Generating Competent *E. coli* Cells by $CaCl_2$ Method

5x M9 salts was composed of 64g/L of disodium phosphate (Na_2HPO_4), 15g/L of monopotassium phosphate (KH_2PO_4), 2.5g/L of sodium chloride (NaCl) and 5g/L of ammonium chloride (NH_4Cl). The final volume was sterilised through a 0.22µm filter.

Minimal agar plates were prepared using 39mL of LB agar heated up to melt completely then left to cool to lower than 50°C. This was combined with 10mL of 5x M9 salts, 1mL of 20% D-glucose, 50µl of 2mg/mL thiamine, 5µl of 1 M $CaCl_2$, and 100µl of $MgSO_4$. The 50mL mixture was subsequently poured equally into 4 separate plates and left to cool in a laminar flow.

5.55g and 6.02g of calcium chloride and magnesium sulfate were weighed and dissolved in 50mL of distilled water to make 1M $CaCl_2$ and 1M $MgSO_4$ respectively. Both solutions were passed through a 0.22µm filter and store in 50mL falcon tubes.

5mL of 1M $CaCl_2$ and 7.5mL of 100% sterilized glycerol were combined, and then topped up to 50mL in a falcon tube to make 0.1M $CaCl_2$ / 15% glycerol. This solution was stored at -20°C, and thaw on ice before use.

Cells were streaked onto minimal agar plates and placed in a static incubator at 37°C overnight. The next day, a colony from this plate was used to inoculate a falcon tube containing 5mL of LB with 100µL of 1M $MgSO_4$. The falcon tube was left to incubate overnight in an incubator shaker.

1mL of the overnight culture was used to inoculate 100mL of pre-warmed LB in a shake flask. The flask was left in an incubator shaker for 2 hours, until the cells were at an optical density of approximately 0.3, the early log phases of the growth curve. At this point, the broth was transferred to chilled, sterile falcon tubes and placed on ice to incubate for 10 minutes. Then

they were subsequently spun at 3300g for 5 minutes in a bench top centrifuge at room temperature. The supernatant was decanted, and any residual liquid was removed carefully using a pipette.

The remaining pellet was resuspended in 10mL of ice cold $CaCl_2$ / 15% glycerol and left to incubate on ice. After a 30 minutes incubation period, the falcon tubes were centrifuged at 3300g for 5 minutes on a bench top centrifuge at room temperature. The supernatant was decanted thoroughly. The pellet in each falcon tube was resuspended in 1mL of ice cold 0.1M $CaCl_2$ / 15% glycerol. The 1mL of competent cells was transferred in 100µl aliquots into pre-chilled Eppendorf, and stored at -80°C.

2.2.5 Transformation into $CaCl_2$ Competent *E. coli* Cells Via Heat Shock

$CaCl_2$ competent cells aliquot were removed from storage and placed onto ice. While on ice, 10µL of plasmid DNA was added. The tube was then left to thaw on ice for 45 minutes. During this incubation period, a water bath was set up at 37°C. After 45 minutes, the eppendorf was transferred into a float in a water bath for 10 minutes, for the process of heat shocking.

The eppendorf tubes were then transferred back onto ice for 2 minutes. After which 1.3mL of LB media with no antibiotic was added into to the competent cells and plasmid mix. The entire volume was then transferred into a 15mL falcon tube and placed into an incubator shaker for 1 hour at 37°C.

The entire volume of the falcon tube was transferred into a new 1.5mL Eppendorf tube and spun at maximum speed in a bench top centrifuge. All the supernatant was carefully removed using a pipette and the remaining pellet was resuspended with 100µl of LB media and spread, in varying volumes, onto agar plate with selection antibiotics: 100µg/mL ampicillin, 50µg/mL

tetracycline and 50μg/mL chloramphenicol. These plates were subsequently left in 37°C static incubator overnight.

2.2.6 Osmotic Shock to Release Periplasm Contents by Disrupting the Outer Membrane

Extraction buffer was composed of 200mM tris hydrochloride (Tris HCl), 1mM disodium ethylenediaminetetraacetate dihydrate (Na_2EDTA), 20w/v sucrose, and 0.5g/L lysozyme.

Overnight cultures were centrifuged at 3300g and the supernatant decanted carefully. The pellet was resuspended in 250μL of extraction buffer and incubated at room temperature. After 15 minutes, 250μL of water was added to induce osmotic shock. The solution was centrifuged at 3300g for 10 minutes and the supernatant was retained for analysis by HPLC.

The volume of extraction buffer and water added to induce osmotic shock was reduced to 100μL at smaller scale cultivation in the TECAN.

2.2.7 Sonication to Release Contents of Cytoplasm and Periplasm

Samples collected in 1.5mL eppendorf tubes were placed in an ice bucket and sonicated using a small probe, with 4 cycles of 10 seconds of sonication followed by 10 seconds of cooling. After sonication, the cells were centrifuged at 3300g for 10 minutes and the supernatant was retained for analysis by HPLC.

2.3 *P. pastoris*

2.3.1 Antibiotics Stock Solutions

Antibiotic stock solution was prepared in 1000X the recommended concentration and was added to culture media in a 1:1000 dilution. Chemicals products were provided by Sigma-Aldrich. Stock solutions of 50mg/mL zeocin were prepared, sterilised by passing through a 0.22µm filter and stored in aliquots of 150 mL in -20 °C. Zeocin is light sensitive, therefore its aliquots were wrap in foil and stored in the dark.

2.3.2 Shake Flasks Cultures for *P. pastoris*

250mL shake flasks were sterilised in an autoclave. 50mL of sterilised culture and the appropriated antibiotic were added to the flasks if required. The media was inoculated with the intended strain and placed in incubator shakers for cultivation for up to 5 days. Any handling of the flasks and media during sampling was handled in a sterilised environment inside a biological safety cabinet.

2.3.3 *P. pastoris* Growth Media

The chemical components of all media were provided by Sigma-Aldrich. Yeast Extract Peptone Dextrose Medium (YPD) was composed of 1% yeast extract, 2% peptone, and 2% dextrose.

Buffered Minimal Glycerol (BMG) was composed of 100mM potassium phosphate, 1.34% YNB, 4 x 10^{-5}% biotin, and 1% glycerol. The media was stored at 4^{o}C. Buffered Minimal Methanol (BMY) and Buffered Minimal Sorbitol (BMS) was made by replacing the glycerol with 0.5% methanol and 1% sorbitol respectively.

Buffered Glycerol-complex Medium (BMGY) was composed of 10g/L yeast extract, 20 g/L peptone, 100mM potassium phosphate, 1.34% YNB, 4 x 10^{-5}% biotin, and 1% glycerol. The media was stored at 4°C. Buffered Methanol-complex Medium (BMMY) and Buffered Sorbitol-complex Medium (BMSY) was made by replacing the glycerol with 0.5% methanol and 1% sorbitol respectively.

2.3.4 Generating *P. pastoris* Competent Cells

Cells were grown in 5mL of YPD in a 50mL falcon tube at 30°C overnight. 0.2mL of the overnight culture was used to inoculate 500mL of fresh YPD and grow overnight again. The culture was aliquoted into 50mL falcons and centrifuged at 1500g for 5 minutes at 4°C. The pellet was resuspended with 500mL of ice cold, sterile water. Cells were centrifuged at the same conditions and resuspended in 250mL of ice cold, sterile water. Cells were centrifuged a third time and resuspended in 20mL of ice-cold 1M sorbitol. The final volume was divided into 80µL aliquots.

2.3.5 Electroporation of *P. pastoris* Cells for Plasmid DNA Uptake

Each aliquot of electrocompetent cells was mixed with 10µg of linearised DNA and transferred to an ice-cold 0.2cm electroporation cuvette from Bio-Rad (Bio-Rad Laboratories, Hercules, California, United States). The cuvette was incubated on ice for 5 minutes. The cells were pulsed using a gene pulser. and 1mL of ice-cold 1M sorbitol was added immediately. The volume was transferred to an eppendorf tube and spread in varying aliquots on YPD plates with selection antibiotics: 50µg/mL zeocin. Plates were incubated at 30°C for two days.

2.4 CHO Cells

2.4.1 Antibiotics Stock Solutions

Antibiotic stock solution was prepared in 1000X the recommended concentration and was added to culture media in a 1:1000 dilution. Chemicals products were provided by Sigma-Aldrich. Stock solutions of 25mg/mL puromycin were prepared, sterilised by passing through a 0.22µm filter and stored in aliquots of 150mL in -20 °C.

2.4.2 Static Cultures for CHO Cells

Disposable sterile Nunc Flasks from Thermo Scientific were used for CHO cell cultures. Cells were seeded in CD CHO defined medium and incubated at 37ºC and 5% CO_2. Cells were passaged when they were of 70% - 80% confluency, approximately 2-3 days, to ensure cell health.

2.4.3 Media for CHO Cell Culture

CD CHO chemically defined medium from Gibco, Thermofisher Scientific, product number 10743029, was used for all CHO cell cultures.

2.4.4 Transfection of DNA via Electroporation

The Amaxa Cell Line Nucleofector Kit V used was procured from Lonza (Lonza Group, Basel, Switzerland). CHO cells were seeded and grown 2-3 days before electroporation. The culture was divided into 1 x 10^6 cells aliquots. Cells were harvested by centrifuging at 100g for 8 minutes at room temperature, and the supernatant was removed completely. The cell pellet was resuspended in 100µL nucleofector solution.

2.5µg of linearised DNA was added to each aliquot. The suspension was transferred into a cuvette and pulsed in the Amaxa. Immediately after, 500µL of culture medium was added to the cuvette. The volume was carefully transferred into a 24-well plate and topped up with culture medium to make a final volume of 1.5mL.

To select for stable transfection, cells were grown for 48 hours in nonselective medium and then transferred to medium containing 25µg/mL puromycin. Cells were placed in humidified incubators at 30^{o}C and 5% CO_2.

2.4.5 Transfection of DNA Using Superfect

The Superfect kit was procured from Qiagen. Cells were split the day before transfection. On the day of transfection, cells were harvested and washed once with PBS. 5 x 10^6 cells were seeded in 60mm dish in 4mL of cell growth medium containing serum and antibiotics.

5ug of DNA was topped up with cell growth medium without serum, proteins, and antibiotics to make a final volume for 150µL. The solution was mixed and spun down to remove drops form the top of the tube. 20µL of superfect reagent was added to the DNA solution, and was mixed by pipetting up and down 5 times.

Cells were incubated at room temperature for 10 minutes. Then 1mL of cell growth medium containing serum and antibiotics was added and mixed. The cells were then immediately transferred to the 60mm dishes. Each dish was gently swirled to distribute the cells.

The cells were then incubated for 48 hours, after which they were washed 4 times with 4mL PBS and passed into medium containing 25µg/mL puromycin.

2.5 Analytical Methods

2.5.1 Gel Protein Electrophoresis

15µL of cell culture sample was combined with 7.5µL of 0.2M dichlorodiphenyltrichloroethane (DDT) and 7.5µl loading buffer and mixed in an eppendorf tube. The tube was placed on a heat block at 95^{o}C for 10 minutes. Pre-casted protein gels from Bio-Rad were placed onto a gel rank inside the gel tank. Running buffered was added to fill the tank to cover the gel. 5µl of marker was loaded alongside the sample cultures in the wells of the protein gel. The gel was run at 200V for 30 minutes.

The gel was removed from the rack and placed in staining reagent for 60 minutes with swing. After which, the staining regent was removed and replaced with water to destain overnight. Gels were placed on a white board and visualized using an illuminator system.

2.5.2 Poros High Pressure Liquid Column Using Agilent Systems

1.38g and 1.42g of sodium phosphate monobasic monohydrate and sodium phosphate dibasic respectively were weighted and dissolved in 1L of distilled water to make 20mM sodium phosphate. The pH of the buffer was adjusted to 7.5 using sodium hydroxide, and the solution was then passed through a 0.22µm filter. This would be referred to as Buffer A.

Another aliquot of 20mM of sodium phosphate was adjusted to the pH of 2.5 using hydrogen chloride. This solution was passed through a 0.22µm filter, and would be referred to as Buffer B.

A HiTrap Protein G was attached to the Agilent (Agilent Technologies, Santa Clara, California, United States) High Pressure Liquid Column (HPLC) system and the column and system were flushed with 20% EtOH and then deionized H_2O for 20 minutes each. The respective lines and

the column were subsequently flushed with Buffer A and Buffer B. During this period, the pumps and lines were checked for air bubbles, and ensured there was no air present before starting analysis. The analytical method for IgG protein was performed as shown in the table below.

Table 2.1: Method design to conduct HPLC analysis using IgG column. In the first four minutes, the sample was introduced to the column and washed with Buffer A. The column was then eluted with Buffer B for 7.5 minutes. After the column was eluted, it was then washed for 2.5 minutes using Buffer A.

Time / Minutes	**Component of Buffer B / %**	**Flow**
0.00	0	1
4.00	0	2
4.01	100	2
11.50	100	2
11.51	0	2
14.00	0	2

300µL of each sample was allocated into individual wells and the sample names are entered corresponding to the well location within the Agilent system sequence table. The sequence was the left to run for its duration.

After analysis, the HPLC and column was flushed with deionized H_2O for 20 minutes and 20% EtOH for another 20 minutes. The column was subsequently removed and stored in -4°C.

2.5.3 Detection of Green Fluorescent Protein using a Spectroflurometer

The spectroflurometer was from Horiba Scientific (Horiba, Kyoto, Japan).

Samples were pelleted and resuspended in phosphate buffered saline (PBS). 1mL of the resuspended cells were pipetted into a clean glass cuvette and placed in the spectroflurometer with excitation wavelength of 395nm and detecting and the emission wavelength from 500nm to 620nm.

2.5.4 Detection of Green Fluorescent Proteins using Flow Cytometer

The Attune NxT Flow Cytometer was from Thermo Fisher Scientific.

Samples were pelleted and resuspended in phosphate buffered saline (PBS). 1mL of the resuspended cells were pipetted into a clean eppendorf and placed in a holder on the cytometer. When using the HT options of the flow cytometer, 300µL of the resuspended cells were pipetted into each well of a 96-well plate. The cytometer was set to analyse 10,000 events with medium flow rate for each sample.

Using a plot of side scatter against front scatter of a negative control, approximately 75% of the events detected were gated. These events were then plotted in a histogram with a marker at approximately 0.5% of the population to account for background fluorescence. These baselines were then used to indicate GFP intensity.

2.5.5 Measurements of Optical Density Using A Plate Reader

The Magellan plate reader was from TECAN (TECAN, Männedorf, Switzerland).

50µL of samples were pipetted into individual wells in a 96-well flat bottom plate. A minimum of 3 columns (24 wells) were pipetted with water as blanks. The Magellan plate reader was set to take multiple readings per well, one in the middle and 4 around the sides as depicted in Figure 2.5.5A. Each reading was taken 5 times at a wavelength of 600nm.

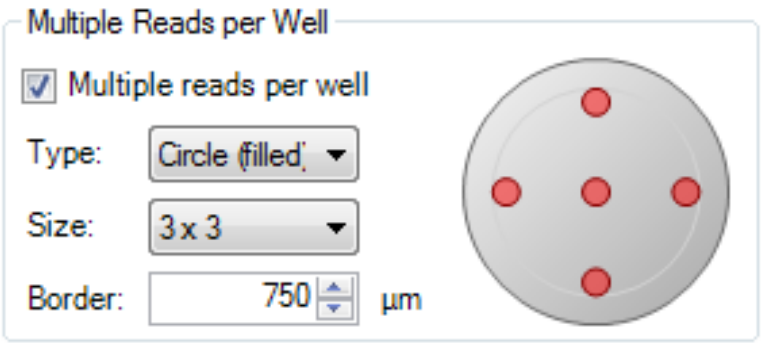


Figure 2.1: Magellan setting for taking measurements at multiple points within a microwell. A reading was taken in the middle of the well, and four were taken around the sides evenly distributed around the border.

2.6 Software Used for Gene Design

2.6.1 A Plasmid Editor for Sequence Editing

A plasmid editor (ApE) maintained by M. Wyane Davis of the University of Utah Biology Department was used for as a sequence editor for gene design. Plasmid maps generated for this project were done using the ApE software.

2.6.2 Codon Optimisation Calculator

The codon optimisation calculator was developed by EnCor Biotechnology Inc. (EnCor Biotechnology Inc., Gainesville, Florida, United States).

2.7 TECAN Worktable Configuration

The TECAN is a laboratory automation work station that is equipped with a liquid handling arm to provide 8 pipetting channels and a robotic arm to transport labware within the worktable. It is used to automate cultivation and sampling at the 800µL scale and preparing samples for the HPLC. The TECAN worktable was set up in the manner depicted in Figure 2.7A for all scripts used in this project.

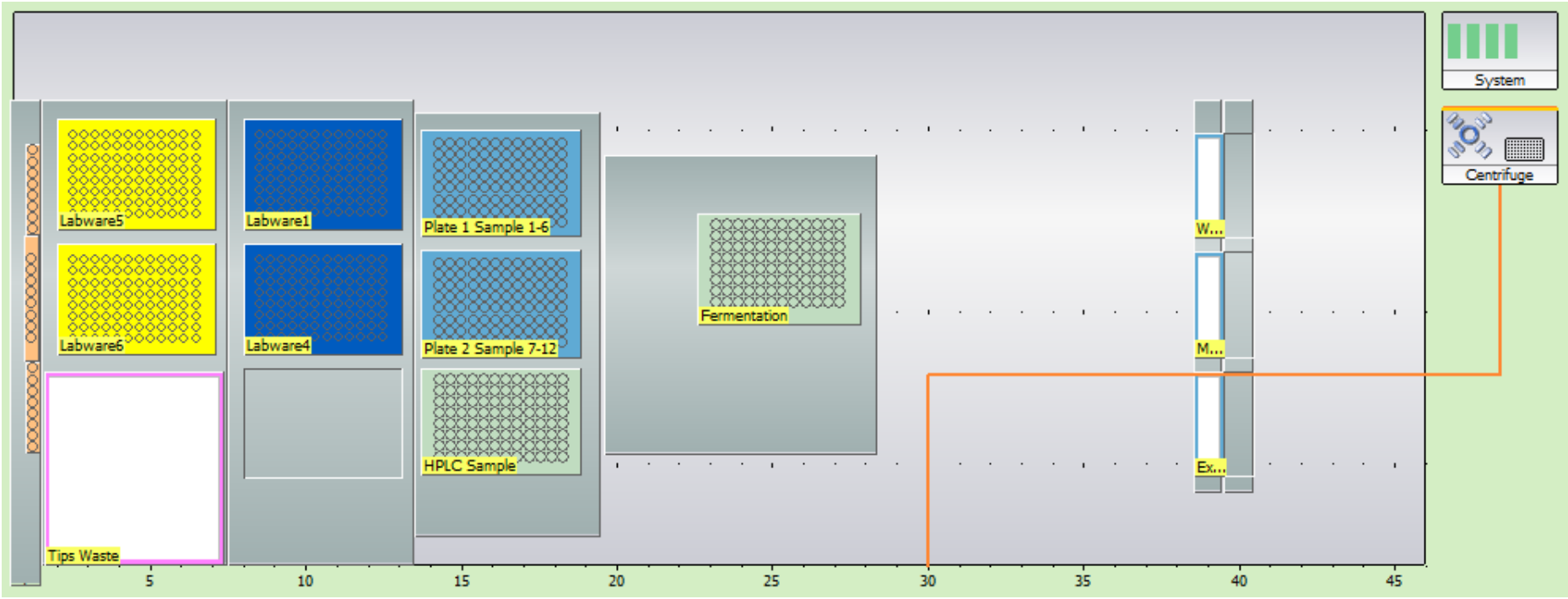


Figure 2.2: Worktable setup of the TECAN. With the liquid handing system wash station at grid 1, 220µL disposable tips carrier and tips waste station at grid 2, 1.2mL disposable tips carrier at grid 8, microplate carrier at grid 14, thermomixer at grid 20, microplate centrifuge at grid 30, and trough carriers at grid 39 and 40. The Magellan plate reader, not depicted in this figure, was placed off the worktable at grid 25.

Chapter 3: A *P. pastoris* Gene Network Designed for Inverse Detection of Methanol

3.1 Concept and Construction of the Inverse Methanol Detector

3.1.1 Repurposing AV4 for Inverse Methanol Detection

Construction of synthetic plasmids is a routine technique employed in molecular biology, yet it is often a time-consuming process (Petit & Petit, 2016). Manipulation of DNA for integration to produce a heterologous protein requires comprehensive understanding of components of gene construction. Therefore, before constructing a completely new plasmid for protein expression, the already synthesized AV4 Syngenta (Syngenta, Basel, Switzerland) plasmid is used to become acquainted with plasmid design considerations, *P. pastoris* handling and transformation techniques. This DNA sequence encodes a vitellogenin protein and is attached to a green fluorescent protein. Fluorescent proteins are effective selective markers, making this construct an ideal starting plasmid.

The fluorescence data gathered from the analysis of this strain was used to develop the software. Allowing the MIAMI software to be able to process fluorescence data from a flow cytometer as well as Fab' data from a HPLC.

3.1.2 Inversed Methanol Induction

The existing plasmid was designed in the manner depicted in Figure 3.1.2A, with the vitellogenin and GFP construct positioned between the gap and AOXI promoter. Once transformed into *P. pastoris*, this sequence will be expressed constituently by the gap promoter independent of carbon source. However, when the AOXI promoter is induced by methanol, it will promote the reserve complimentary RNA of the sequence expressing the proteins. In theory, the RNA encoding the proteins, and its reverse compliment strand will bind to each other, and thereby block protein translation.

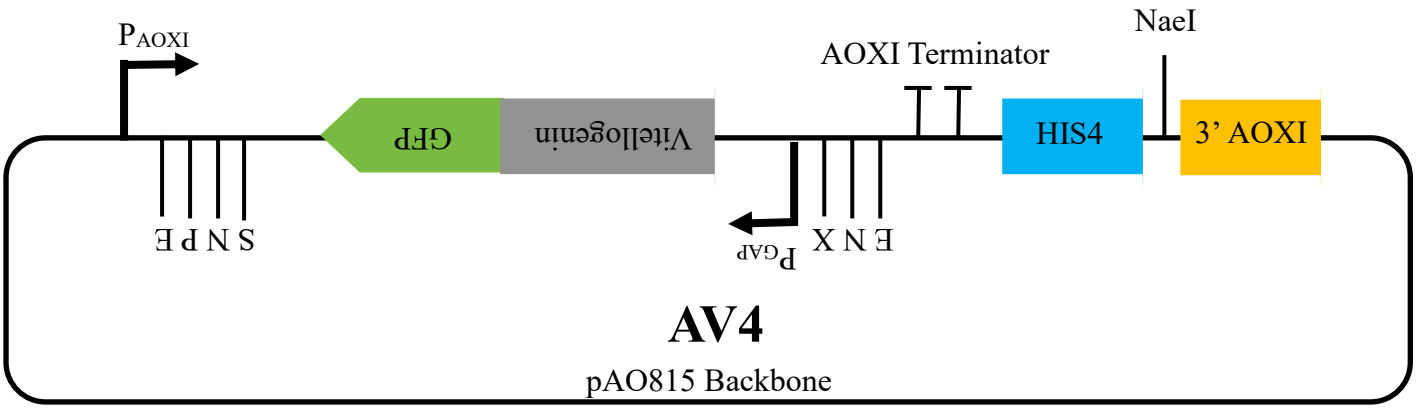


Figure 3.1: Design cassette of the AV4 plasmid, synthesized to express the vitellogenin and GFP protein in *P. pastoris*. GFP is was attached to the end of the vitellogenin sequence, this is promoted by a GAP promoter. On the reverse complimentary strand, an AOXI promoter and terminator was placed at the ends of the GFP vitellogenin sequence, promoting the reverse complimentary of the GFP vitellogenin.

Evaluating the efficiency of a promoter is important, especially when it is used in plasmid designs. When they are used in different express systems, it is challenging to determine the potency of one promoter in relation to another.

3.2 Creation of the AV4 *P. pastoris* Strain

3.2.1 Isolation of the AV4 Plasmid and Transformation

The plasmid was transformed into Top10 bacterial host cells. The pAO815 plasmid possesses bacterial ampicillin resistance. The strain was grown in LB and plasmid was purified from the resulting overnight media.

Potential linearisation sites for integration into the GS115 yeast genome were BglII, SalI, StuI, and BspEI. A review of the AV4 sequence showed that all these restriction sites were present within the protein sequence and thus were not viable. A unique restriction site, NaeI, was selected for linearization. This site did not cleave through any of the synthesized protein DNA sequence and was situated between the HIS4 and 3'AOXI features. The linearised plasmid was transformed in electrocompetent GS115 via electroporation.

3.2.2 Confirming GFP in *P. pastoris* Cells

Ten colonies were picked from the petri dishes and grown in YPD, along with wild type GS115 for control. The cells from the overnight cultures were analysed using a spectroflurometer. Out of the ten colonies screened, only one indicated GFP presence. The results from the spectroflurometer are shown in Figure 3.2.3A. For clarity of the graph, the results from only one unsuccessful transformation is included. Its excitation profile behaved identically to the control. Whereas, the successful transformation displayed higher excitation at 500-520nm. This wavelength of light is observed as green, which corresponds to the expected colour exhibited by the GFP.

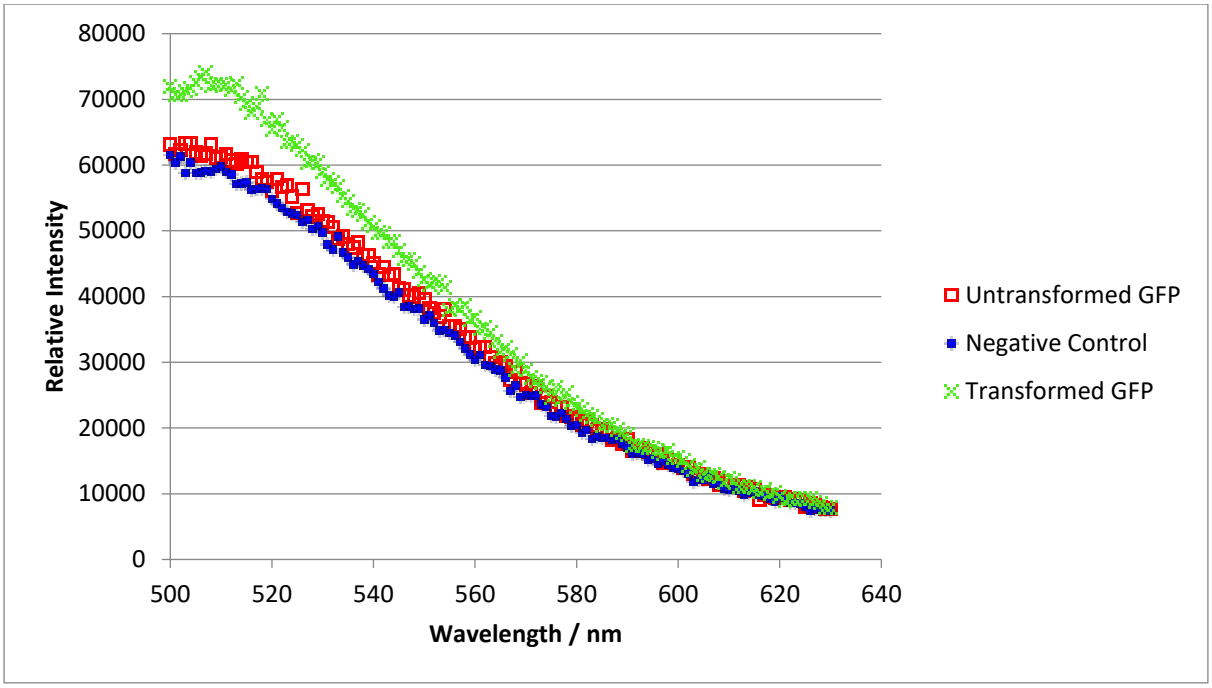


Figure 3.2: Results from the spectroflurometer, showing the presence of GFP when excited by light of different wavelength. The unsuccessfully transformed GFP behaved in a similar profile to the negative control, decreasing in intensity as the wavelength increased. The transformed GFP expressed higher intensity between the shorter wavelengths of 500nm to 520nm when compared to the negative control. There appeared to be a peak in intensity at approximately 510nm for the transformed GFP, with its intensity decreasing as the wavelength increased.

The same control and overnight cells were passed through the flow cytometer. Once the control was gated as shown in Figure 3.2.3B (Top), the same colony transformation displayed GFP presence, whereas the remaining nine did not. The consistent results confirm that this colony had been successfully transformed with the AV4 insert. The GS115 strain transformed with the AV4 insert would be referred to as GS-AV4 for the purpose of this thesis.

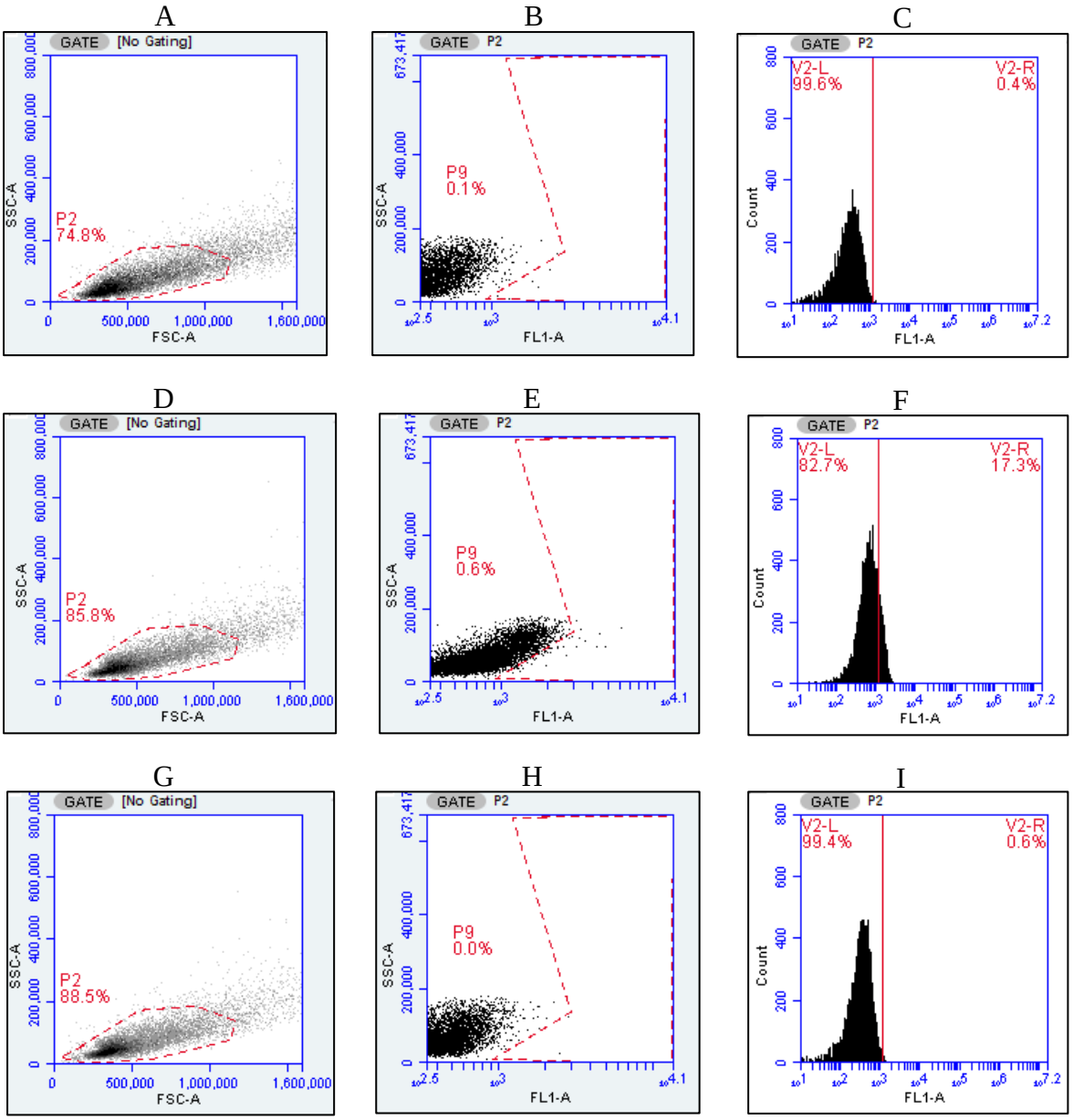


Figure 3.3: GFP measurements in *P. pastoris* cells. Negative Control (A, B & C), Successful Transformation (D, E & F), and Unsuccessful Transformation (G, H & I). The graph in A, the negative control, is gated for population analysis. The gated population is displayed as a density graph in C, a marker at 0.4% to set to account for background fluorescence. Using the same maker, 17.3% of the population in F exhibits GFP, indicating a successful transformation. Whereas in in I, 0.6% of the population exhibits GFP, suggesting this was an unsuccessful transformation.

3.3 Characterisation of the Inverse Methanol Sensor with Different Carbon Sources

The use of methanol carbon source would inhibit the production of the integrated protein, hence switching-off the green fluorescence in the cells. In the interest of validating this, GS-AV4 was grown in 250mL shake flasks over a period of 96 hours using different media with different carbon sources. After 24 hours of initial fermentation, the carbon sources would be changed by pelleting the culture using centrifugation and then resuspending in fresh media comprised of a different carbon source. In doing so, it not only mapped the growth and behaviour of the strain in different media, but also displayed how quickly the switch mechanism reacts when the new carbon source is introduced.

3.3.1 Shake Flask Cultures Using Glycerol and Methanol Complex Media

As shown in Figure 3.3.1A, GS-AV4 grown in glycerol complex media reached a maximum GFP expression at approximately 24 hours and maintained that maximum throughout the remainder of the growth period. The same strain in methanol complex media maintained a constant minimal GFP production throughout the growth period. Similarly, wild type GS115 grown in the same methanol media did not produce any GFP as well.

A medium switch from glycerol to methanol resulted in an immediate cessation of GFP production. The medium switch from methanol to glycerol did not instantaneously increase the GFP production. After a period of approximately 24 hours, GFP was produced and rapidly reached a steady maximum. The 24 hours delay appeared to coincide with the 24 hours lag period when the strain was grown in glycerol. Consequently, once the medium switch was changed to glycerol, the productivity profiles both followed a similar trend.

This page is intentionally left blank.

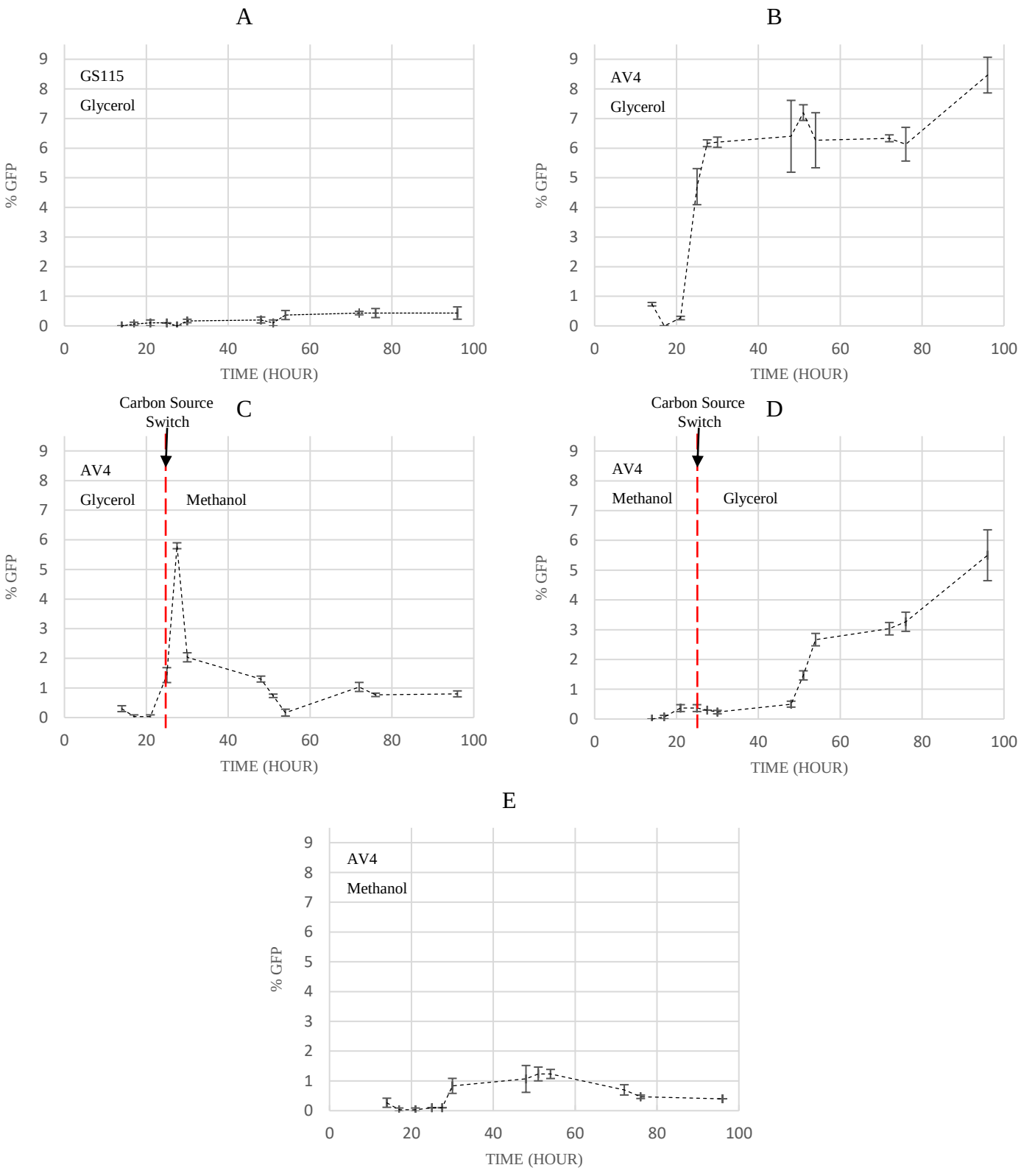

A
GS115
Glycerol
% GFP
TIME (HOUR)
B
AV4
Glycerol
% GFP
TIME (HOUR)
C
Carbon Source Switch
AV4
Glycerol
Methanol
% GFP
TIME (HOUR)
D
Carbon Source Switch
AV4
Methanol
Glycerol
% GFP
TIME (HOUR)
E
AV4
Methanol
% GFP
TIME (HOUR)

Figure 3.4: Compilation of graphs showing the percentage of culture population expressing GFP and their associate error bars over a period of 96 hours shake flask culture. (A) is the negative control, with GS115 grown using glycerol carbon source. While percentage of cells expressing GFP increased as time passed, it was very marginal, remaining comfortably under 1.0 even with the increasing error margins throughout the entire cultivation period. (B) shows GS-AV4 grown using glycerol carbon source. Percentage of cells expressing GFP remained low at the initial hours of fermentation. After approximately 24 hours, this increased dramatically from under 1.0 to over 6.0 and maintained this level of expression throughout the remainder of the 96-hour cultivation period. (C) shows GS-AV4 grown initially with glycerol carbon source for 24 hours. The carbon source was then swapped to methanol. The red dotted line represents the point at which the carbon source was switched. Percentage of cells expressing GFP remained low at the initial hours of fermentation. After approximately 24 hours, this increased dramatically and reached a peak of almost 6.0. After the peak, percentage of cells expressing GFP decreased rapidly and remained low at around 1.0 for the remainder of the 96-hour cultivation period. (D) shows GS-AV4 grown initially with a methanol carbon source for 24 hours. The carbon source was then swapped to glycerol. The red dotted line represents the point at which the carbon source was switched. Percentage of cells expressing GFP remained low for approximately 50 hours at under 1.0. After which this increased to 3.0 and maintained it, with GFP expression steadily rising to 5.5 at the end for the 96-hour cultivation period. (E) shows GS-AV4 grown using methanol carbon source. Percentage of cells expressing GFP remained low throughout the 96-hour cultivation period, with a mild peak at approximately hour 50. However, this perceived peak was still of low at 1.2.

3.3.2 Sorbitol's Effect on the Inverse Methanol Sensor in Shake Flask Cultures

Sorbitol is an alternative carbon source to glycerol that can be used in conjunction with methanol for regulating the AOXI promotor. Glycerol yields higher volumetric productivity with lower aeration requirement (Berrios, et al., 2017); while sorbitol induced lower osmotic stress, resulting in less host cell protein leakage (Shen, et al., 1999). Both are commonly used carbon source for *P. pastoris* fermentation, hence it is important to determine how the novel AV4 system performs in sorbitol medium.

As observed from the results plotted in Figures 3.3.1A-E, the effects of the carbon source switch can be perceived by the end of a 2-day, 48-hour, period. Keeping the time of media swap consistent with the pervious shake flask cultures, the observation period for GFP intensity was shortened to 48 hours.

This page is intentionally left blank.

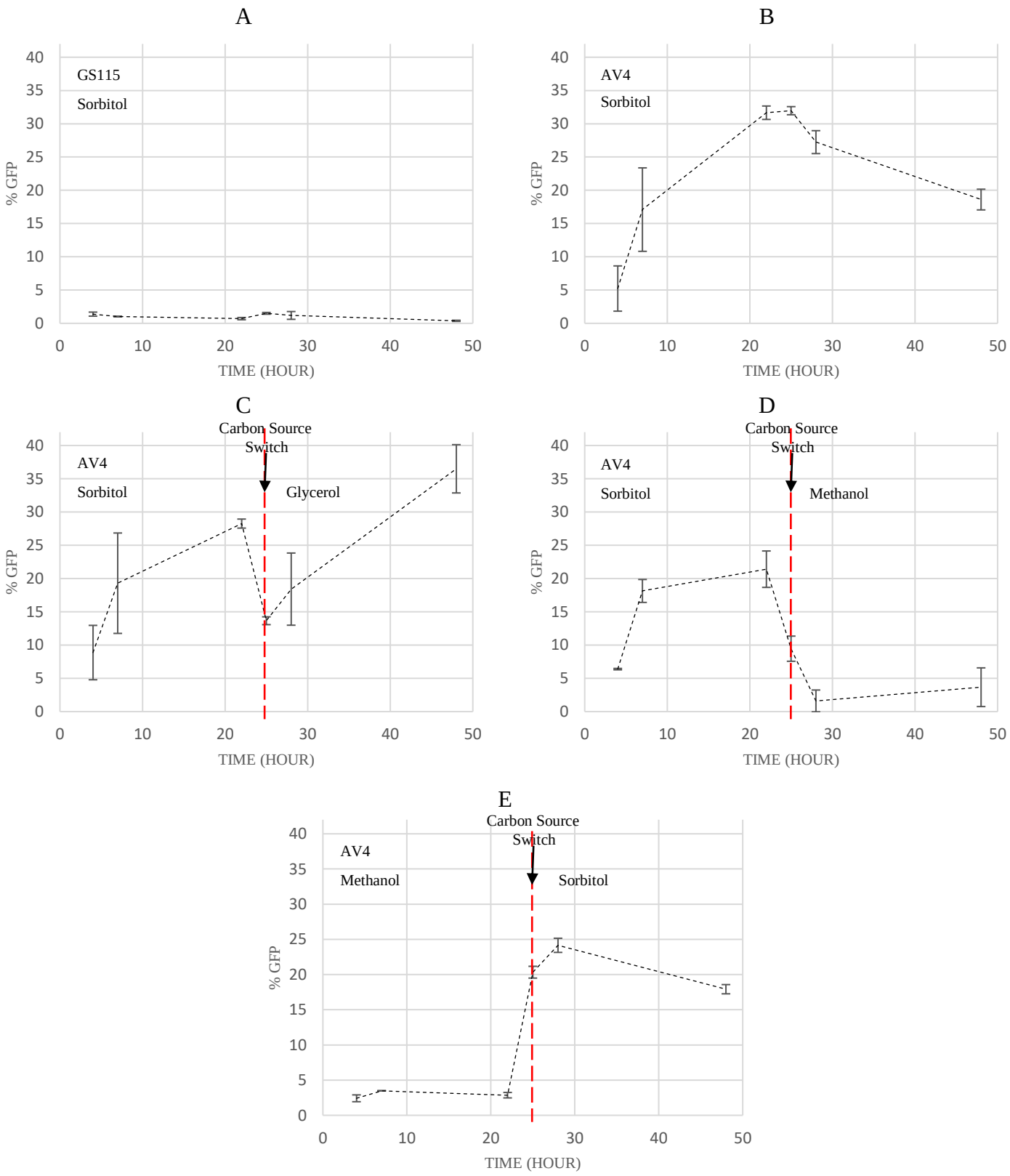

A
GS115
Sorbitol
% GFP
TIME (HOUR)
B
AV4
Sorbitol
% GFP
TIME (HOUR)
C
Carbon Source Switch
AV4
Sorbitol
Glycerol
% GFP
TIME (HOUR)
D
Carbon Source Switch
AV4
Sorbitol
Methanol
% GFP
TIME (HOUR)
E
Carbon Source Switch
AV4
Methanol
Sorbitol
% GFP
TIME (HOUR)

Figure 3.5A: Compilation of graphs showing the percentage of culture population expressing GFP and their associated error bars over a period of 48 hour shake flask culture. (A) is the negative control, with GS115 grown using sorbitol carbon source. Percentage of cells expressing GFP remained low throughout the entire 48-hour cultivation period. (B) shows GS-AV4 grown using sorbitol carbon source. Percentage of cells expressing GFP increased exponentially and reached a peak of over 30.0 after 24 hours. After the peak, this decreased slowly for the remainder of the 48-hour cultivation period. (C) shows GS-AV4 grown using sorbitol carbon source for 24 hours before the carbon source was swapped to glycerol. The red dotted line represents the point at which the carbon source was switched. Percentage of cells expressing GFP increased logarithmically but experienced a temporary drop immediately after the carbon source swap. However, this increased logarithmically for the remainder of the 48-hour cultivation period. (D) shows GS-AV4 grown using sorbitol carbon source for 24 hours, at which point the carbon source was switched to methanol. The red dotted line represents the point at which the carbon source was switched. Percentage of cells expressing GFP increased logarithmically. There is an immediate drop after the carbon source swap and remained low for the remainder of the 48-hour cultivation period. (E) shows GS-AV4 grown using methanol carbon source for 24 hours, at which point the carbon source was swapped to sorbitol. The red dotted line represents the point at which the carbon source was switched. Percentage of cells expressing GFP remained low but experienced a dramatic increase immediately after the carbon source swap and maintained a relatively high percentage GFP expression for the remainder of the 48-hour cultivation period.

As depicted in Figure 3.5, GS-AV4 behaved similarly in sorbitol and glycerol up to approximately 20 hours. However, GS-AV4 maintained a steady GFP production for longer in glycerol. Both sorbitol and glycerol should not inhibit the production of GFP. Therefore, when the medium was switched from sorbitol to glycerol at 24 hours, the cells maintained GFP production despite a mild dip immediately following medium change. The temporary depression was likely a lag period in cell growth when the cells were recovering from the shock of the medium change.

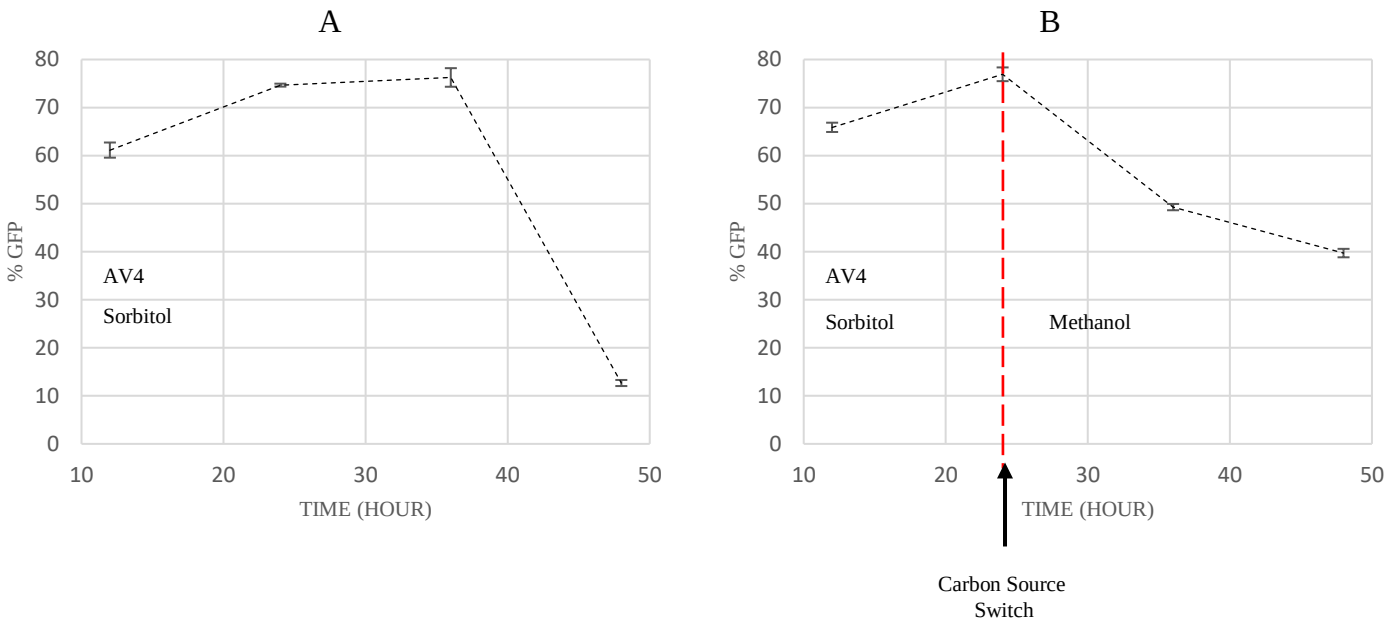
The medium change from sorbitol to methanol resulted in the immediate disappearance of GFP, behaving in the exact manner as in the medium change from glycerol to methanol. As expected, GFP was detected when the medium was switched from methanol to sorbitol. Unlike glycerol, sorbitol seemed to start producing GFP more rapidly, with no lag period observed in the results.

It must be noted that there was a change in flow cytometry used to analyse the results between the data presented in section 3.2 and 3.3. This was due to an unforeseen fault with the BD Accuri^{TM} C6 Plus Flow Cytometer. The equipment could not be repaired in a timely manner, therefore Attune NxT Flow Cytometer was used for the remainder of the project. Although this might limit the comparison that can be drawn between the two separate data sets, it does not invalidate the conclusions of the strains' characteristics within the data sets individually.

3.3.3 Characterisation of the Inverse Methanol Sensor in Minimal Media

Complex media is designed for general use that promotes rapid growth, however, the components of each batch of media are unknown and can vary. To demonstrate that the behaviour of the switch mechanism is dependent solely on the carbon source, rather than interaction between the undefined components, minimal medium is used for the same media switch productivity profiling.

Since the production of GFP in sorbitol appeared without a lag period, unlike in glycerol, the growth in minimal media was monitored for a shorted period of 48 hours, with a sample taken every 12 hours. The results are plotted and shown in Figure 3.3.3A.


A
% GFP
AV4
Sorbitol
TIME (HOUR)
B
% GFP
AV4
Sorbitol
Methanol
TIME (HOUR)
Carbon Source
Switch

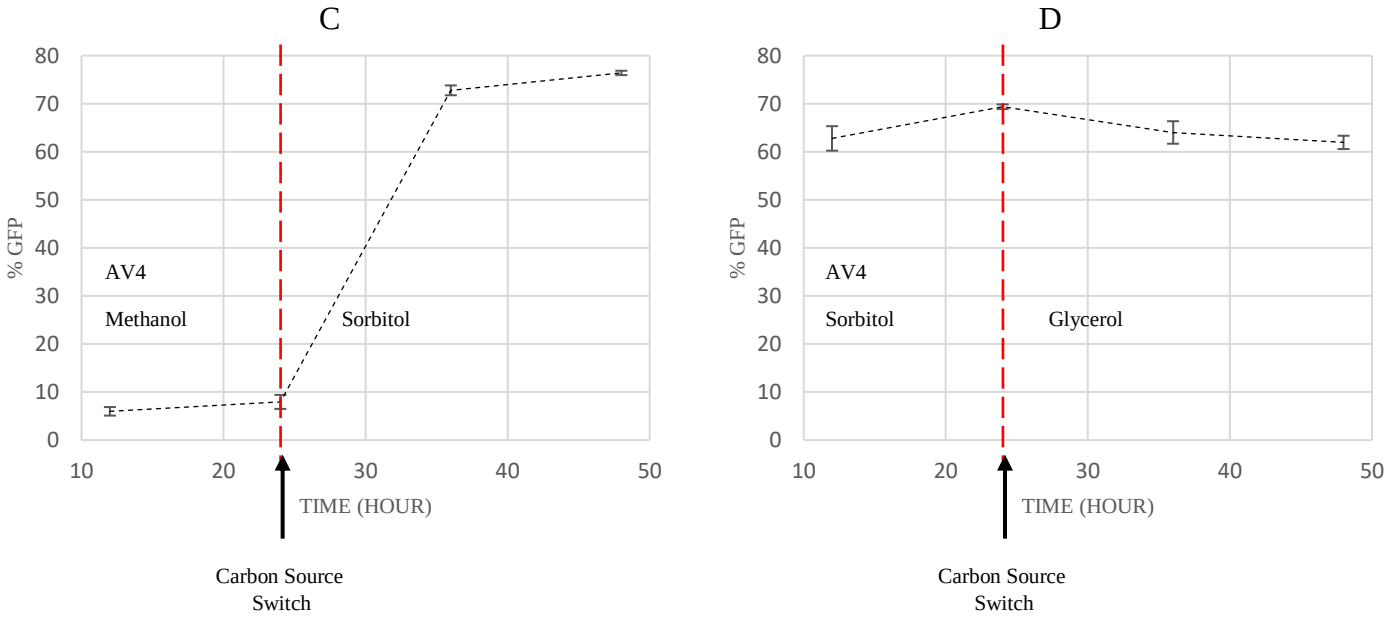

C
% GFP
AV4
Methanol
Sorbitol
TIME (HOUR)
Carbon Source
Switch
D
% GFP
AV4
Sorbitol
Glycerol
TIME (HOUR)
Carbon Source
Switch

Figure 3.6: Compilation of graphs showing the percentage of culture population expressing GFP and their associated error bars over a period of 48 hour shake flask culture using minimal media. (A) show the GS-AV4 grown using minimal sorbitol media. The percentage of cells expressing GFP held a constantly high value but decreased after 36 hours. (B) shows GS-AV4 grown initially using minimal sorbitol media, then switched to minimal methanol media after 24 hours. The red dotted line represents the point at which the carbon source was switched. Percentage GFP expression held a consistently high value for the first 24 hours. After the media switch, it decreased gradually for the remainder of the 48-hour cultivation period. (C) shows GS-AV4 grown initially using minimal methanol media. This was switched to minimal sorbitol media after 24 hours. The red dotted line represents the point at which the carbon source was switched. Percentage expression of GFP held a consistently low value for the first 24 hours. After the media switch, GFP intensity increased dramatically and held a consistently high value for the remainder of the 48-hour cultivation period. (D) shows GS-AV4 grown initially using minimal sorbitol media, then switched to minimal glycerol media after 24 hours. The red dotted line represents the point at which the carbon source was switched. Percentage expression of GFP held a consistently high value for the entire of the 48-hour cultivation period.

In comparison to complex medium, minimal medium was shown to be able to achieve double the percentage of cells expressing GFP. Reaching approximately 70.0 instead of 32.0. The error bars are smaller and more consistent in minimal medium. This can be attributed to the defined components of the medium used. This highlights the importance of defined medium components in reducing variability. By comparing these results to those in the previous section, it can be concluded that the switch mechanism in the AV4 strains performed in the same manner in minimal and complex medium. This suggests with relative certainty that the use of an AOXI promotor as an inverse methanol sensor is working as intended.

3.4 Chapter Conclusion

The AV4 strain functions as intended. GFP signals are detected when cultivating using glycerol and sorbitol as a carbon source. However, GFP signals are always inhibited by the presence of methanol.

Chapter 4: Design of Plasmid Insert and Transfection into CHO Cells

4.1 Creating Data Sets for Use in the Development of MIAMI

A protein expression system was designed spanning three commonly used host cells in the production of biotherapeutics: *E. coli*, *P. pastoris*, and CHO. A typical protein expression system was created by bearing in mind the considerations a conventional researcher would contemplate, such as whether the Fab' is secreted or whether the Fab' is produced constituently. This provided a typical strain for each host cell type. They were used to demonstrate the cell line selection capabilities of the MIAMI software.

These three typical strains will be used to create a data set for use in developing the MIAMI software. The ranking algorithm of the cell lines would be developed using the knowledge-based evaluation of the same data set. With this approach, the data sets highlighted the important factors that needed to be prioritised in the development of MIAMI. Basing the algorithms on real information significantly aided the MIAMI development process.

4.2 Identifying an IP-Free Sequence for MIAMI Development Data Sets

The design of the expression systems was adapted to coincide with recommendations of relevant literature. Therefore, a genomic sequence without intellectual property claims was used to create the expression systems for this research, ensuring the freedom to modify the genomic sequence as required without repercussions. For this reason, great effort was invested in determining viable intellectual property free sequences for use.

Although there were other protein sequences easily available, and a search showed that there were no intellectual property claims on them currently. However, there is no guarantee that their intellectual property free status will remain for the duration of this project. Thus, a sequence with an expired patent would be more suited for the intended purposes. Taking into consideration that patents are usually granted for a period of 10-15 years from the date of application, a systematic search was conducted for patents involving protein sequences that were filed 20 years prior. The additional 5 years was a grace period to ensure that the patent was not extended, thus allowing for the freedom to use and modify the sequence.

A systematic search through an online registry (www.expiredpatents.com) of expired patents filed before 1994 yielded a patent for the expression of Anti-Hepatitis B antibody. The online registry used is no longer running. This is one of the three antibodies that binds to hepatitis B virus, was filed in 1992 and expired by 2004 (Kurihara, et al., 1992). It includes the full sequence, heavy and light chains of the antibody. The research conducted as part of the application revealed that the antibody had been successfully expressed in both prokaryotic and eukaryotic host cells. Similar patents for this sequence were published under different applications, however all of them had expired a decade ago. Assuming the expiration of the patent symbolises the expiry of intellectual property claims of the anti-hepatitis B sequence, this was used for the development of a typical protein expression system.

4.3 Designing the Truncation of the Anti-Hepatis B Sequence to Create a Fab' Fragment

An intended secondary purpose of this project was to draw a more direct comparison between the three different host strains. Comparing advantages and limitations between host strains is often based on the expression of completely different products. By expressing the exact same protein in *E. coli*, *P. pastoris*, and CHO cells, there should be a clearer indication of which hosts excel in which aspect.

While CHO cells are less restricted by the expressed protein size and have complex mechanisms with the ability to synthesis, process, secrete, and glycosylate proteins, *E. coli* is less capable of expressing larger proteins (Yin, et al., 2007). Since the typical heterologous protein needs to be expressed by *E. coli*, the antibody sequence was altered to instead produce an anti-hepatitis B Fab' fragment. The locations of the disulphide bonds were identified using cysteine sites present in the sequence and the antibody was truncated 3 basepairs after the disulphide bonds present at the hinge of the antibody.

The locations of the disulphide bonds were identified using the cysteines sites. The pattern of the disulphide bonds suggested the antibody is of IgG_1 subtype (Liu & May, 2012). The IgG subtype is further confirmed through identifying the location of glycosylation sites by searching for their known motifs (Valliere-Douglass, et al., 2010; Spiro, 2002). The glycosylation motifs were only in the C_H2, and C_H3 regions of the antibody, which is a trait unique to IgG subtypes.

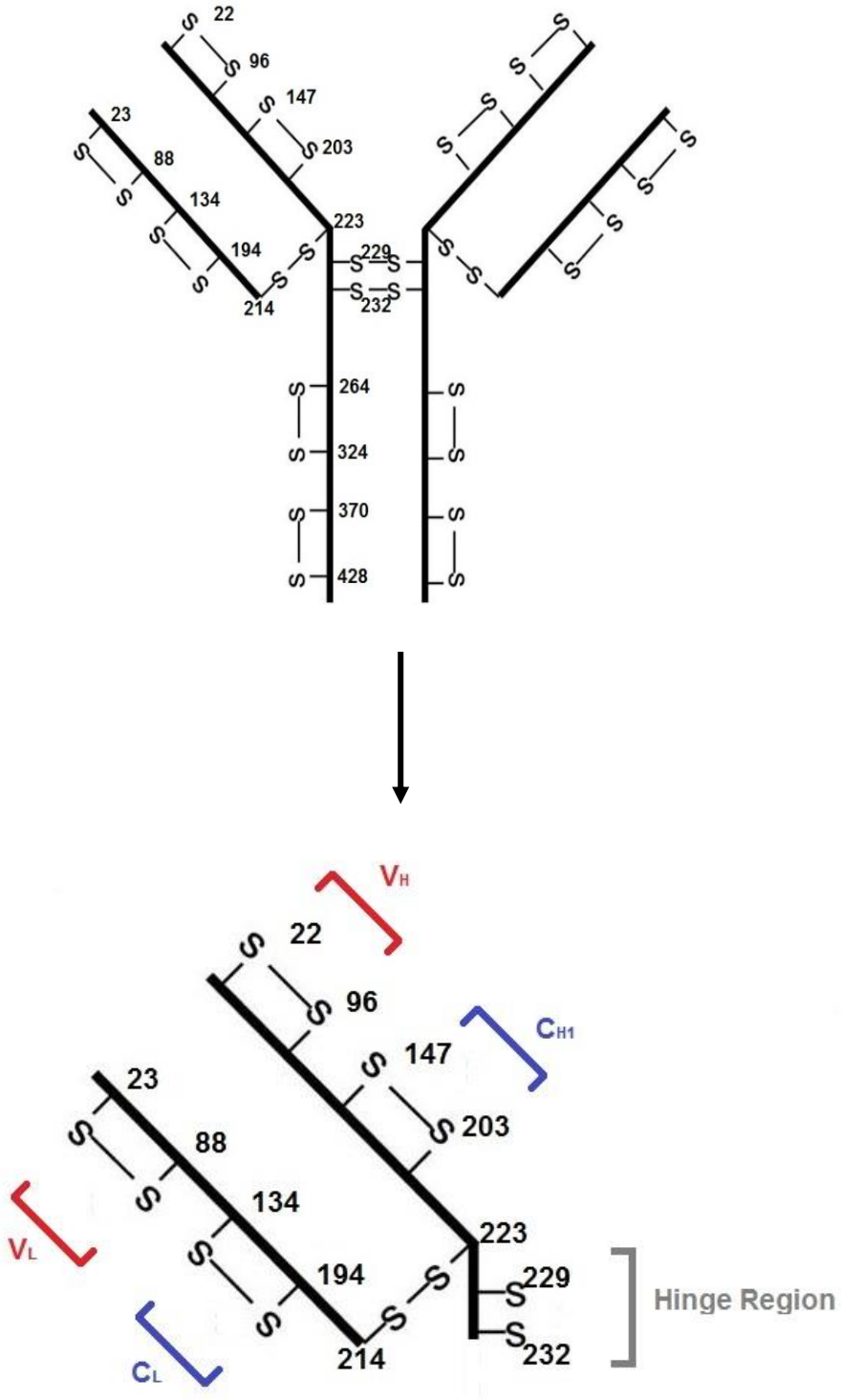


Figure 4.1: Diagram of the anti-hepatitis B antibody (Top) and the anti-hepatitis B Fab' fragment (Bottom). The disulphide bonds are located and depicted at 22, 96, 147, 203, 229,232, 264, 324, 370 and 428 basepairs on the heavy chain, and at 23, 88, 134, 194 and 214 basepairs on the light chain. The heavy chain of was truncated three base pairs after the hinge region of the antibody, at 235 basepairs to create the anti-hepatitis B Fab' fragment.

4.4 Design of the Fab' Expression Cassette for Mammalian Cells

The components used in the design for the insert into CHO cells were from two different plasmids, pSecTag2A and pIRESpuro. Homologous recombination occurs when there are similar or identical DNA sequences. Repeated DNA would increase the likelihood of double strand breaks. To avoid this, two different promoters and terminators were used in the insert as shown in Figure 4.3A. The designed sequence was inserted into pUC57 through the flanking unique restriction sites, XbaI and BamHI. This high copy number plasmid was used to produce multiple copies of this sequence which can be subsequently used to transfect into CHO cells.

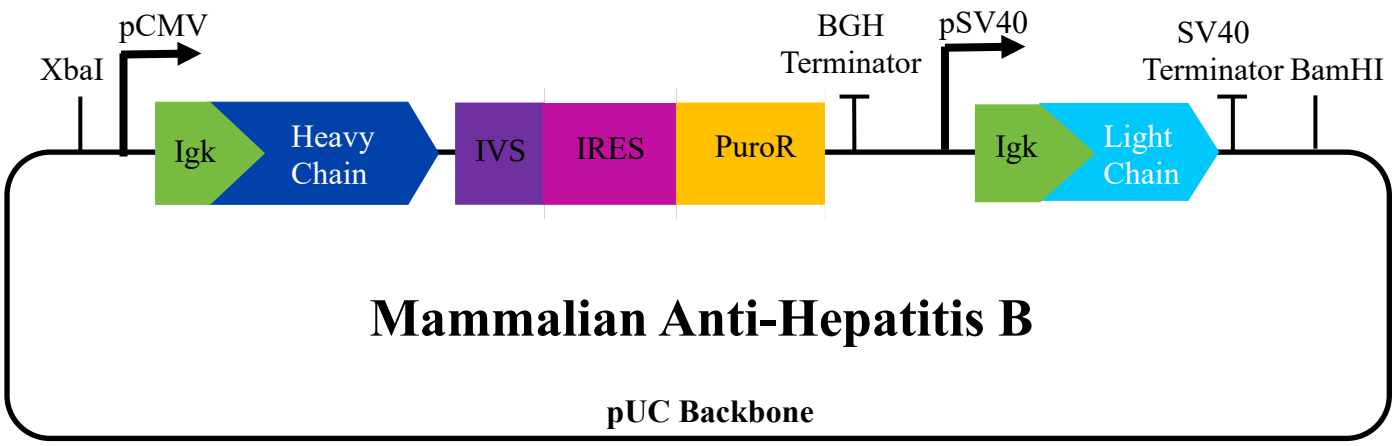


Figure 4.2: Expression cassette for the anti-hepatitis B Fab' fragment for mammalian cells. The heavy chain of the Fab' fragments is preceded by an IgK leader, and followed by synthetic intron (IVS), internal ribosome entry site (IRES), puromycin resistance (PuroR). The sequence containing the heavy chain is placed between a CMV promoter and BGH polyadenylation terminator. The light chain of the Fab' fragment is preceded by an IgK leader and placed between a SV40 promoter and SV40 terminator. This entire insert is placed between the unique restriction sites XbaI and BamHI of a pUC57 backbone. The pUC57 backbone contains an ampicillin resistance gene and a ColE1 origin of replication.

4.5 Transfection into Adherent and Suspension CHO Cells

There are two types of transfection: transient and stable. In transient transfection, the nucleic acids introduced to the CHO cell are not permanently incorporated into the cellular genome, therefore the protein will only be expressed for a short period of time (Yin, et al., 2007).

To permanently establish foreign DNA, it must be integrated into the chromosomes of the CHO host cells. Once it has been integrated into the cell's genome, it can constitutively and permanently express the protein (Kunert, et al., 2008).

4.5.1 CHO Cell Transfection Attempt Using Electroporation

2.5µg of circular mammalian anti-hepatitis B plasmid was used to transfect GS-CHO. A plasmid expressing a green fluorescent protein, pGTIP and water were also transfected into the CHO cells as a positive and negative control respectively. After a two-day incubation period, the CHO cells for the positive control appeared green, which suggested a successful transfection. The anti-hepatitis B plasmid transfected CHO cells were resuspended into media containing puromycin. After an incubation period of 8 weeks, all the transfected CHO cells died at the same rate as the negative control, indicating the plasmid did not incorporate into the CHO cells genome. The unsuccessful transfection of the anti-hepatitis B plasmid is likely due to quality or quantity of the plasmid used in the transfection. Since the CHO cells in the positive control expressed GFP, CHO cell quality or the electroporation protocol is unlikely to be the cause of failed transfection.

4.5.2 CHO Cell Transfection of CHO Cells Using SuperFect

GS-CHO null cells were transfected with 2.5µg of circular mammalian anti-hepatitis B plasmid using SuperFect. The same cells were also transfected with water as a negative control. All transfected cells were left to incubated for 2 days at normal growth conditions. Subsequently,

puromycin was added to the media as a selective medium. After a period of 6 weeks, there appeared to be a population of healthy cells growing in the selective medium.

Although the foreign DNA was established, there was an unforeseen equipment failure causing the loss of the transfected cells. The time lost as a result meant the development of this cell line was halted in favour of developing the *E. coli* and *P. pastoris* cell lines.

4.6 Chapter Conclusion

The IP-free mAb sequence was truncated to create the anti-hepatitis B Fab'. The mammalian insert was cloned into a pUC backbone. The plasmid was purified and transfected into suspension CHO cells using SuperFect.

Chapter 5: Designing and Expressing Fab' Expression in *E. coli*

5.1 Fab' Expression in *E. coli*

The original anti-hepatitis B sequence from the patent was optimised for mammalian cell expression. The sequence was truncated to create a Fab' fragment. The smaller size allowed for expression in *E. coli*. However, other adaptations were implemented to optimise the expression in *E. coli* host cells and account for its limitations.

5.1.1 Locating Possible Glycosylation Sites in the Anti-Hepatitis B Sequence

E. coli hosts lack the cellular machinery to perform post-translational modifications such as glycosylation, leading to unfolded proteins that are incapable of performing their intended functions. While renaturation can provide the protein its appropriate three-dimensional shape, it leads to issues in downstream processing such as reduced yields or formation of aggregates. Taking into consideration the additional workload these processes would place on downstream processing, it would be advantageous to reduce or remove glycosylation sites within the sequence.

Disulphide bonds in the original full antibody were identified by the location of the cysteine sites present in the sequence. The pattern of the disulphide bonds suggested that the antibody was of the IgG_1 subtype (Liu & May, 2012). Glycosylation sites are usually only present in the C_H2 and C_H3 region of IgG antibodies (Verma, et al., 1998). Further search of the sequence revealed no known motifs of glycosylation sites in the V_H1, C_H1, V_L and C_L regions (Valliere-Douglass, et al., 2010; Spiro, 2002). As a result, it can be assumed with high certainty that the truncation of the sequence to create an antibody Fab' fragment removed all glycosylation sites.

5.1.2 Optimisation of Codon Usage for *E. coli* Expression

Redundancies exist within the translation of mRNA; Different triples of codons translate as the same protein. Codon usage biases are present in all eukaryotic and prokaryotic genomes, where the use of preferred codons correlates with higher expression (Zhou, et al., 2016). For expression of the Anti-Hepatitis B Fab' in *E. coli* hosts, the codons for both the light and heavy chains were optimised for bacterial hosts with a codon optimising calculator constructed by Encor Biotechnology Inc.

5.1.3 Designing Biobrick Insert Sites and Silent Mutations

Four biobrick sites were introduced to the sequence: EcoRI-HF and XbaI preceding the sequence and SpeI and PstI at the end. These sites were included to offer the sequence to a library of standardized components for open source usage. For these sites to be utilized as intended, they must be unique. A search of the full sequence was performed to identify the aforementioned biobrick sites. Any duplicates were removed from the sequence using silent mutations, preserving the translated sequence while removing the restriction site.

Table 5.1: Table showing the location of the biobrick restriction sites present in the original Fab' sequence. The original and alter sequences both translate to the same protein.

Restriction Site	Location	Original Sequence (5' -3')	Original Protein	Altered Sequence (5'-3')	Altered Protein
Heavy Chain					
EcoRI	4-9	GAA TTC	Glu	GAG TTC	Glu
PstI	325-330	CTG CAG	Leu	CTC CAG	Leu
PstI	598-603	CTG CAG	Leu	CTC CAG	Leu
Light Chain					
PstI	160-165	CTG CAG	Leu	CTC CAG	Leu
PstI	232-237	CTG CAG	Leu	CTC CAG	Leu
PstI	460-465	CTG CAG	Leu	CTC CAG	Leu
PstI	625-630	CTG CAG	Leu	CTG CAG	Leu

5.2 Design Fab' Expression Cassette for *E. coli*

5.2.1 Low Copy Number Plasmid Choice

With protein propagation as a priority, a low copy number plasmid was more desirable since it would not place unwarranted stress on the host cell to replicate multiple copies of the plasmid. An intellectual property free plasmid was used in conjunction with the intellectual property free Fab'. pACYC184 is a plasmid with a copy number of approximately 15 with two antibiotic resistance genes which were useful for antibiotic selection. The plasmid map is shown in Figure 5.1.

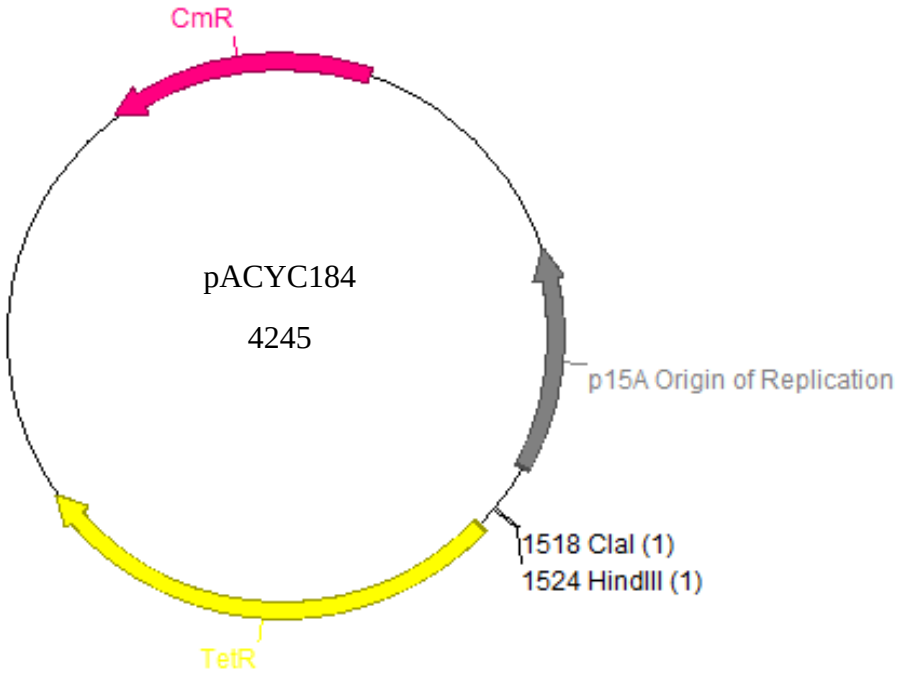


Figure 5.1: Plasmid map of pACYC184. This 4245 basepair plasmid has chloramphenicol resistance, tetracycline resistance and p15A origin of replication. Also depicted on the plasmid map are the unique restriction sites ClaI and HindIII. They are located between the origin of replication and tetracycline resistance. The anti-hepatitis B Fab' sequence would be inserted between these two sites.

5.2.2 Promoter and Secretion Signal Choice

An OmpA secretion signal was attached before both the sequence of the heavy and light chain. Secretion of the Fab' into the periplasm had two tangible benefits. It allowed for the correct disulphide bonds to form in the oxidizing environment and accounted for easier purification steps as there were less contaminating host cell proteins in the periplasm. Less stress is required to lyse only the outer membrane of the host cells, resulting in less fractionated cell membrane particulates that causes fouling of filtration units in downstream processing.

A tacI promotor was added upstream of both the OmpA secretion signal. This hybrid promoter can be repressed by the lac repressor, which can be subsequently unrepressed using isopropyl beta-D-thiogalactoside (IPTG) (de Boer, et al., 1983). This function would prove to be useful for controlling expression of this foreign gene in its *E. coli* host.

5.2.3 Transcription Terminators Choice and Location

Terminators were placed immediately after both the heavy chain and light chain. This ensured the termination of protein translation and avoided excess amino acid sequences attached to the resulting Fab'. Two different terminators were used in the design of this plasmid insert to avoid lengths of repeated DNA. The two available terminators were: BBA_B001, a double terminator, and BBa_B1002, a single terminator.

The required spacing between the stop codon, TAA, and the terminator itself depends on the terminator and strain. A review of past literature suggests that transcription terminators are generally found 133 base pairs downstream of the stop codon (de Hoon, et al., 2005). For bidirectional promotors, the distance between promotors facing opposite direction is approximately 240-300 base pairs apart (Shu, et al., 2006; Takai & Jones, 2004). Spacers created from randomly generated DNA were placed strategically within the sequence in

accordance to the literature. Different sequences for spacers were used within each design to avoid repetition of DNA.

5.2.4 Plasmid Insert Design Options

Three different designs were created for consideration as shown in Figure 5.2, Figure 5.3 and Figure 5.4. Promoters, secretion signals, spacers and terminators were placed using the rationale discussed in the previous sections. These designs were placed between two unique restriction sites in the plasmid, ClaI and HindIII.

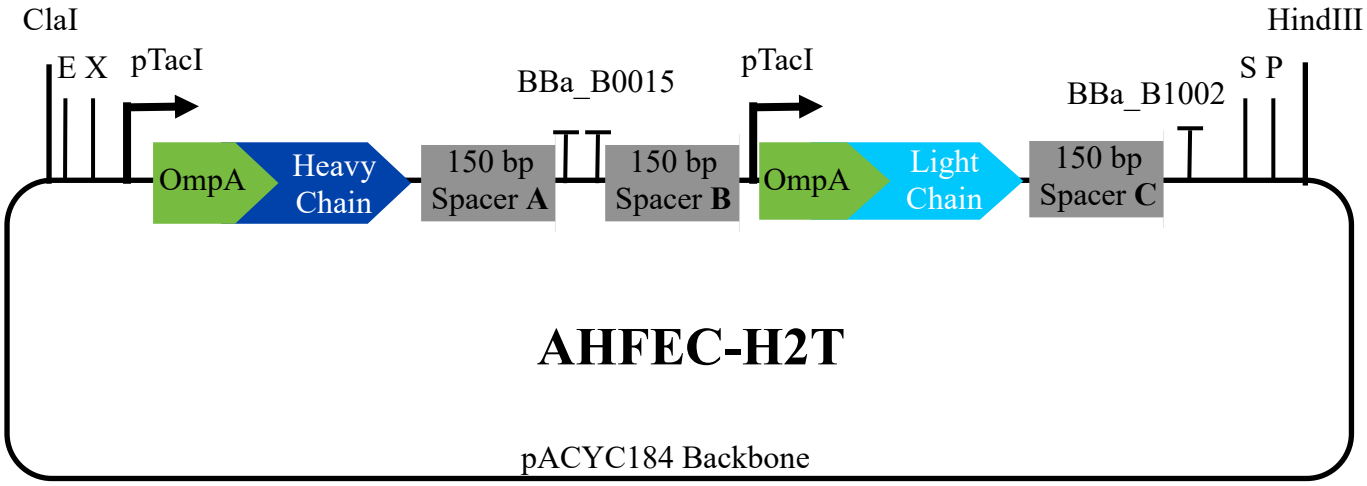


Figure 5.2: Design cassette α showing the orientation and spacing of each component, lab code name AHFEC-H2T. A pTacI promoter and OmpA secretion signal were placed upstream of both the heavy and the light chain of the Fab' fragment. Downstream of both sequences were 150 bp spacers and the terminators BBa_B0015 and BBa_B1002 respectively. The sequences were orientated head to tail, promoting in the same direction. A 150 bp spacer was placed between the BBa_B0015 terminator for the heavy chain and the pTacI promotor for the light chain. The insert was placed between the unique restriction sites ClaI and HindIII on the pACYC184 plasmid backbone.

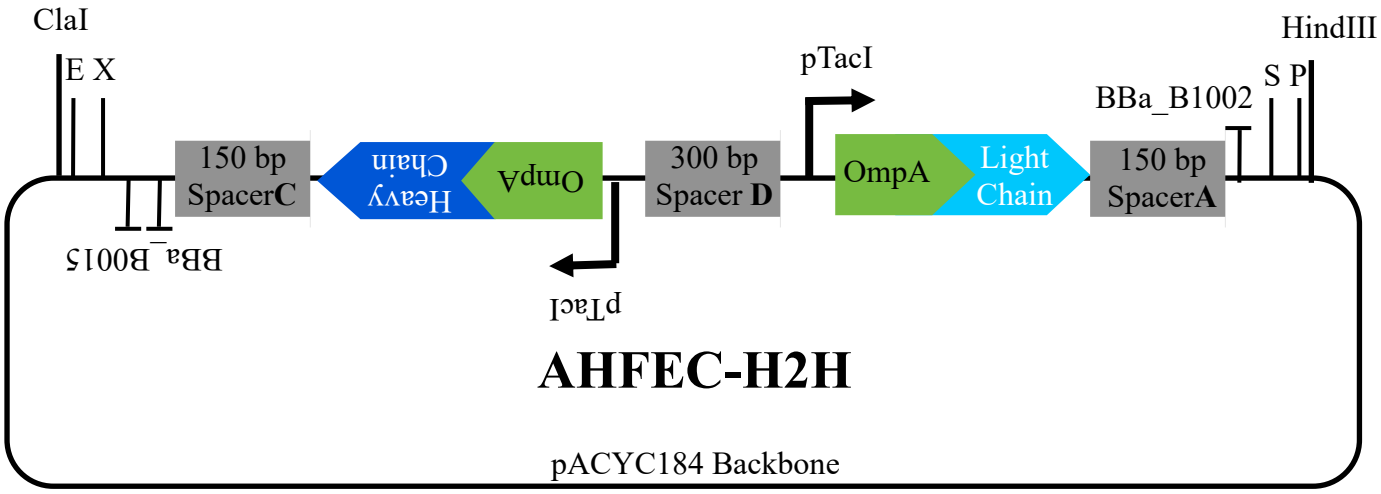


Figure 5.3: Design cassette β showing the orientation and spacing of each component, lab code name AHFEC-H2H. A pTacI promoter and OmpA secretion signal were placed upstream of both the heavy and the light chain of the Fab' fragment. Downstream of both sequences were 150 bp spacers and the terminators BBa_B0015 and BBa_B1002 respectively. The sequences were orientated head to head, promoting in opposite directions, away from each other. A 300 bp spacer was placed between the two pTacI promoter. The insert was placed between the unique restriction sites ClaI and HindIII on the pACYC184 plasmid backbone.

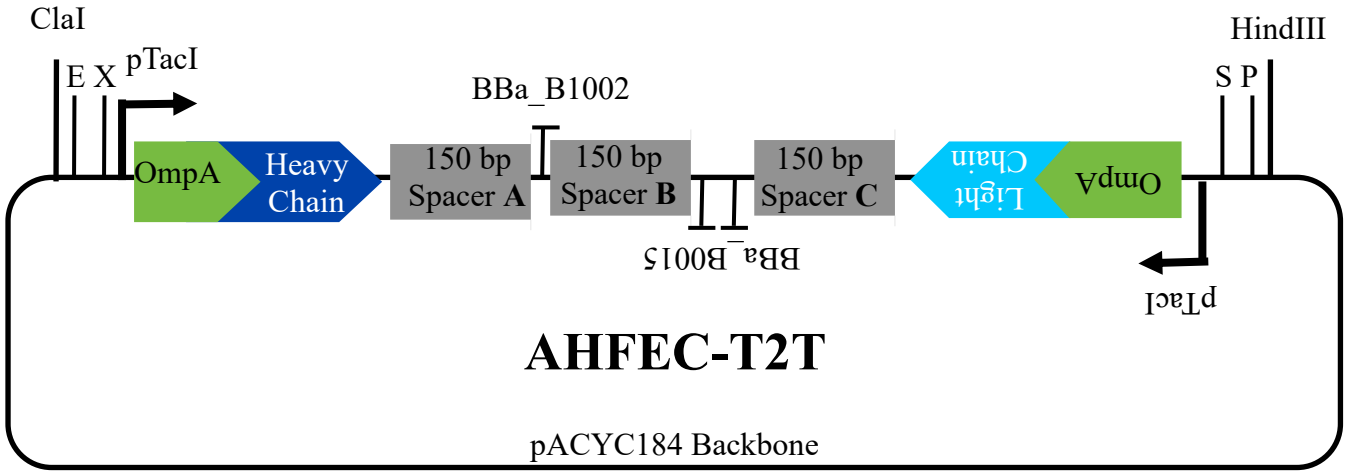


Figure 5.4: Design cassette γ showing the orientation and spacing of each component, lab code name AHFEC-T2T. A pTacI promoter and OmpA secretion signal were placed upstream of both the heavy and the light chain of the Fab' fragment. Downstream of both sequences were 150 bp spacers and the terminators BBa_B1002 and BBa_B0015 respectively. The sequences were orientated tail to tail, promoting in opposite directions towards each other. A 150 bp spacer was placed between the two terminators separating them. The insert was placed between the unique restriction sites ClaI and HindIII on the pACYC184 plasmid backbone.

Design cassette α as, shown in Figure 5.2.4A, was best suited as both chains were orientated in the same direction, both the heavy and light chains would be produced in equal amounts, making it an efficient choice. If the chains were orientated in opposite directions, as in design cassette β and γ, the chain encoded in one direction might be produced in favour of the chain encoded in the opposing orientation.

One concern regarding the design of in Figure 5.2.4A was the possibility that transcription of the heavy chain would not be fully terminated and may have included parts of the light chain sequence. This would have unknown effects on the translation of the sequence. However, the use of spacers between each chain, in conjunction with the double terminator, should ensure the effective termination of the heavy chain.

5.3 Generating the AHFEC-H2T *E. coli* Strain for Fab' Expression

The insert depicted in Figure 5.2 was synthesized in collaboration with Eurogentec. The insert was then cloned into pACYC184, between the tetracycline resistance gene and the origin of replication, via the ClaI and HindIII unique restriction sites. The synthesised plasmid was transformed into Top10 *E. coli* strain.

5.3.1 Cloning Strategy of Expression Cassette α (AHFEC-H2T)

The original design intended for the constant production of Fab'. The tacI promotor, without the presence of a lac repressor, will be continuously producing Fab'. As the first synthesized plasmid for a foreign host, it was important to verify thoroughly that it performed as intended. An existing strain created by UCB is IPTG inducible, therefore a Lac repressor was introduced to the AHFEC-H2T strain. In making it inducible by the same mechanic, it would allow for direct comparison to be drawn between the two strains and allow for a better verification of its validity.

The LacI sequence was present in the pMMB66EH plasmid. This produces the Lac repressor, which binds to the tacI promotor in the AHFEC-H2T plasmid and inhibits the transcription until it is relieved by lactose or IPTG. Both the pMMB66EH and AHFEC-H2T plasmid was transformed into the same host cell to create an IPTG inducible strain that produced the anti-HBS Fab' fragment.

The *E. coli* strains available were, W3110 and Top10. $CaCl_2$ competent cells of both strains were created through heat shock, ready for transformation. Both plasmids were isolated and purified from their respective strains. 10µg of AHFEC-H2T and pMMB66EH were used for transformation into the prepared competent cells. Both plasmids must be present in the transformed strain, therefore the transformation was selected for antibiotic resistance to ampicillin, tetracycline and chloramphenicol. This generated colonies of W3110 and Top10 cells with both pMMB66EH and AHFEC-H2T plasmids, named QT-WAHT66 and QT-TAHT66 respectively.

Table 5.2: Table summarising the names of the *E. coli* strains used, their host cells and the plasmids they contain. The name listed here would be used to refer to each strain throughout the remainder of this thesis.

Name	**Host Cell**	**Plasmids**	
		AHFEC-H2T	pMMB66EH
QT-TAHT	Top10	✓	X
W-MMB	W3110	X	✓
QT-WAHT66	W3100	✓	✓
QT-TAHT66	Top10	✓	✓

5.3.2 Creating a Standard Curve to Analyse HPLC Data

Running a known concentration of pure Fab', 0.02g/L, 0.1g/L, 0.2g/L, 0.3g/L, and 0.4g/L, in the HPLC allows for the creation of a standard curve. Plotting the concentration against the elution peak area provides an equation that can be used to process raw HPLC data into Fab' concentration. Figure 5.5 shows the plot where the standard curve was derived from. Equation (1) shows how to convert elution peak area to Fab' concentration.

$$Fab' Concentration = Elution\ Peak\ Area \times 0.00003 + 0.0144 \quad (1)$$

Since the y-intercept of the equation is not at zero, the minimum Fab' concentration would be 0.0144g/L. This would pose a problem when dealing with low elution peak areas because the 0.0144g/L increment might affect the shape of the productivity profile. Thus, when processing raw HPLC data, if no peak was detected, the Fab' concentration will default to 0g/L, rather than derived using Equation (1).

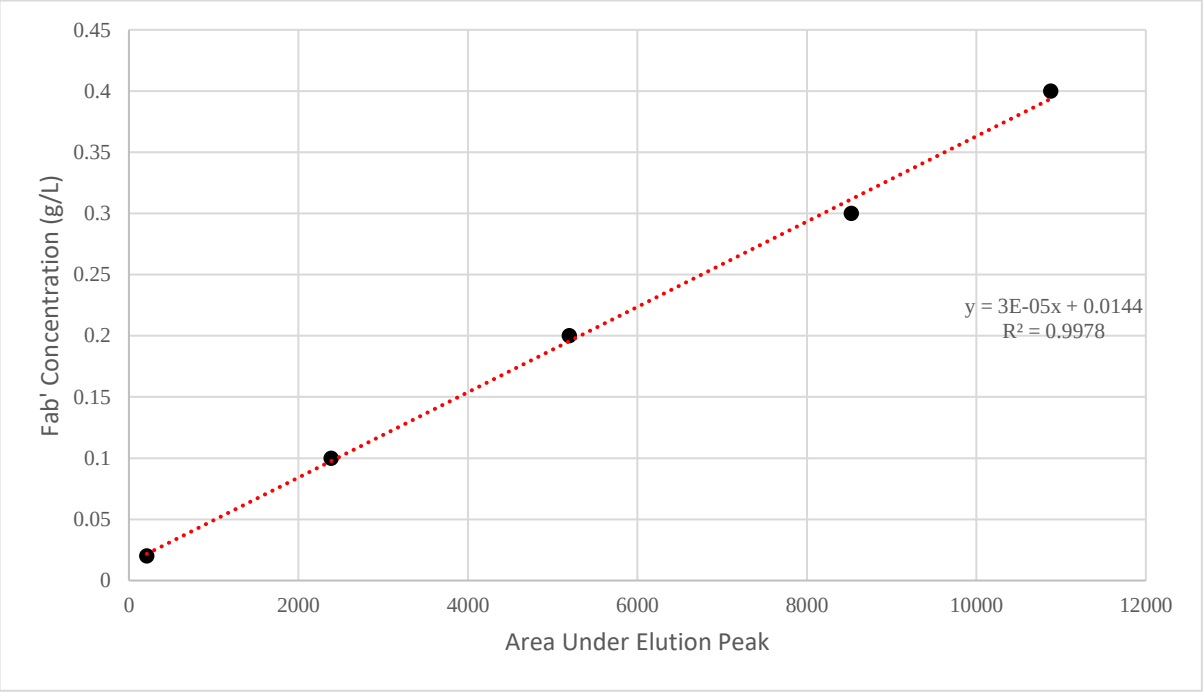


Figure 5.5: Plot of known Fab' concentration against the area of their respective elution peaks. A linear equation is fitting to the data sets and shown as a red dotted line.

5.3.3 Confirming Successful Transformation into *E. coli* Cells

When QT-WAHT66 and QT-TAHT66 were grown alongside the UCB strain, they all performed in a similar manner. Initially, cells were grown using shake flasks in 50mL of LB. While HPLC results indicated the successful production of Fab', Fab' was produced even in uninduced samples for all three strains. After some investigation, it appeared that lactose was present in variable quantity in tryptone, a primary component of LB media. The lactose prevented the lac repressor from inhibiting transcription of the sequence encoding the product. Resulting in the production of Fab' whether the cells were induced or not. Since this irregularity was present in both newly synthesized strains and the well-established UCB strain, it was likely a symptom of media composition rather than error in the synthetic plasmid design.

To overcome this issue, the cells were grown in SM6G defined media following a predefined protocol for the optimised growth of the UCB Fab'. It did not use tryptone as an ingredient and therefore eliminated the issue of lactose presence causing premature induction of the Fab' product. The strains were initially grown in 50mL of 2XPY complex media, which used peptone instead of tryptone in its composition. After 3-4 hours of growth, when OD was around 1, 1mL of the 2XPY media was used to inoculate 50mL of SM6G media in 250mL shake flasks. The flasks were then grown in normal conditions for 12 hours and subsequently monitored on a bihourly basis.

Using this new protocol, the UCB strain behaved as intended, however, the QT-WAHT66 and QT-TAHT66 strains still produced Fab' when uninduced. Upon further reflection, this might have been a result of the tacI promotor and the lac repressor being present on separated plasmids. The difference in copy numbers of the host plasmids suggested that there might have been a lack of lac repressor produced for the number of AHFEC-H2T plasmids containing the tacI promotor. Full profiles of all the HPLC results are available in Appendix I.

Table 5.3: Amount of Fab' produced per litre in uninduced and IPTG-induced samples for the three *E. coli* strains: TAHT66, WAHT66, and UCB.

Strain Name	**Host Strain**	**Uninduced Sample (mg/L)**	**Induced Sample (mg/L)**
QT-TAHT66	Top10	34.34	35.68
QT-WAHT66	W3110	27.69	31.75
UCB	W3110	0.00	46.37

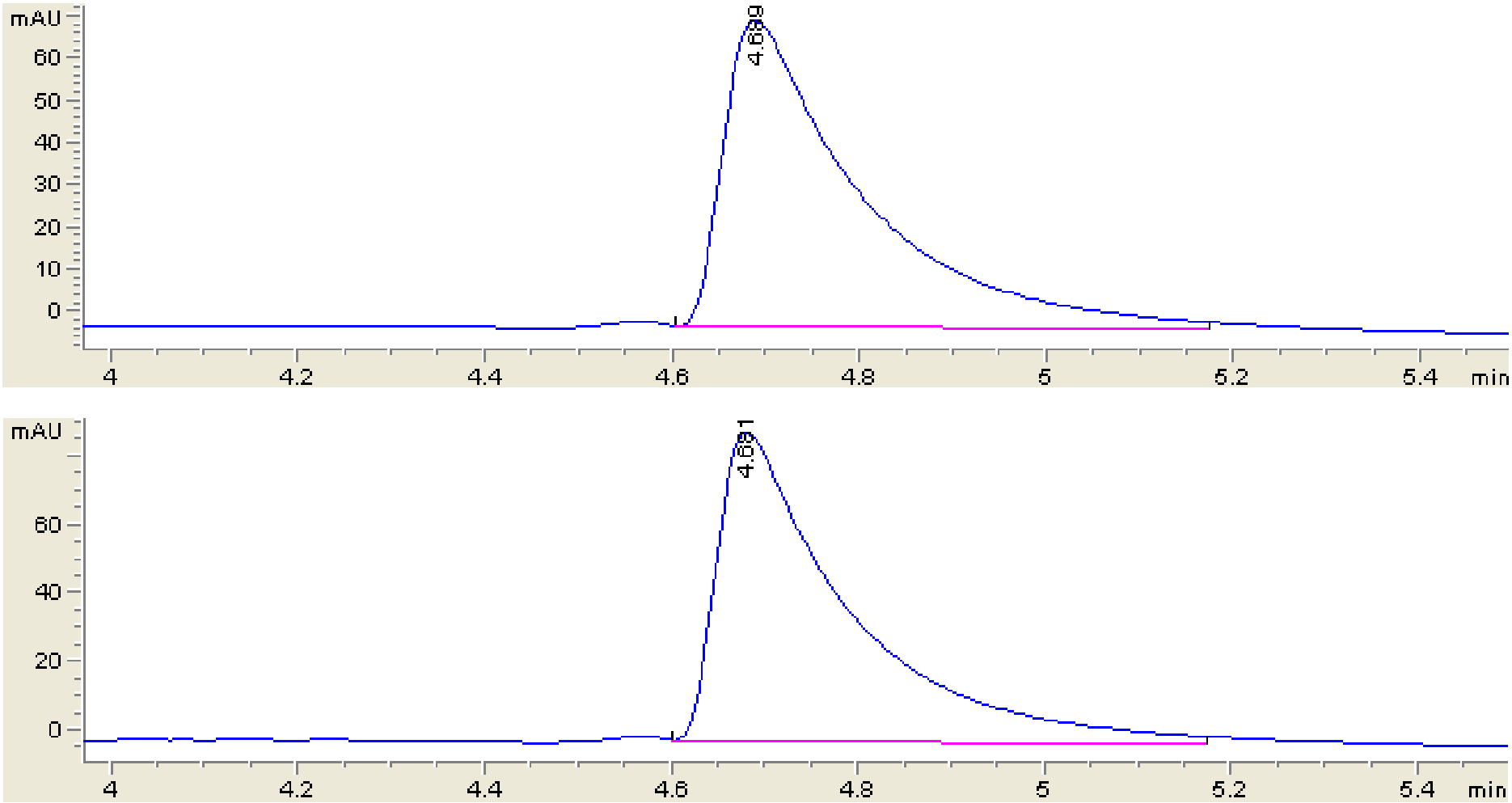


Figure 5.6: Excerpts of HPLC results for uninduced (Top) and induced (Bottom) samples of QT-TAHT66. There were elution peaks for uninduced and induced QT-TAHT66 samples, with corresponding Fab' concentration of 34.34mg/L and 35.68mg/L respectively. Full profiles of both HPLC results are available in Appendix I.

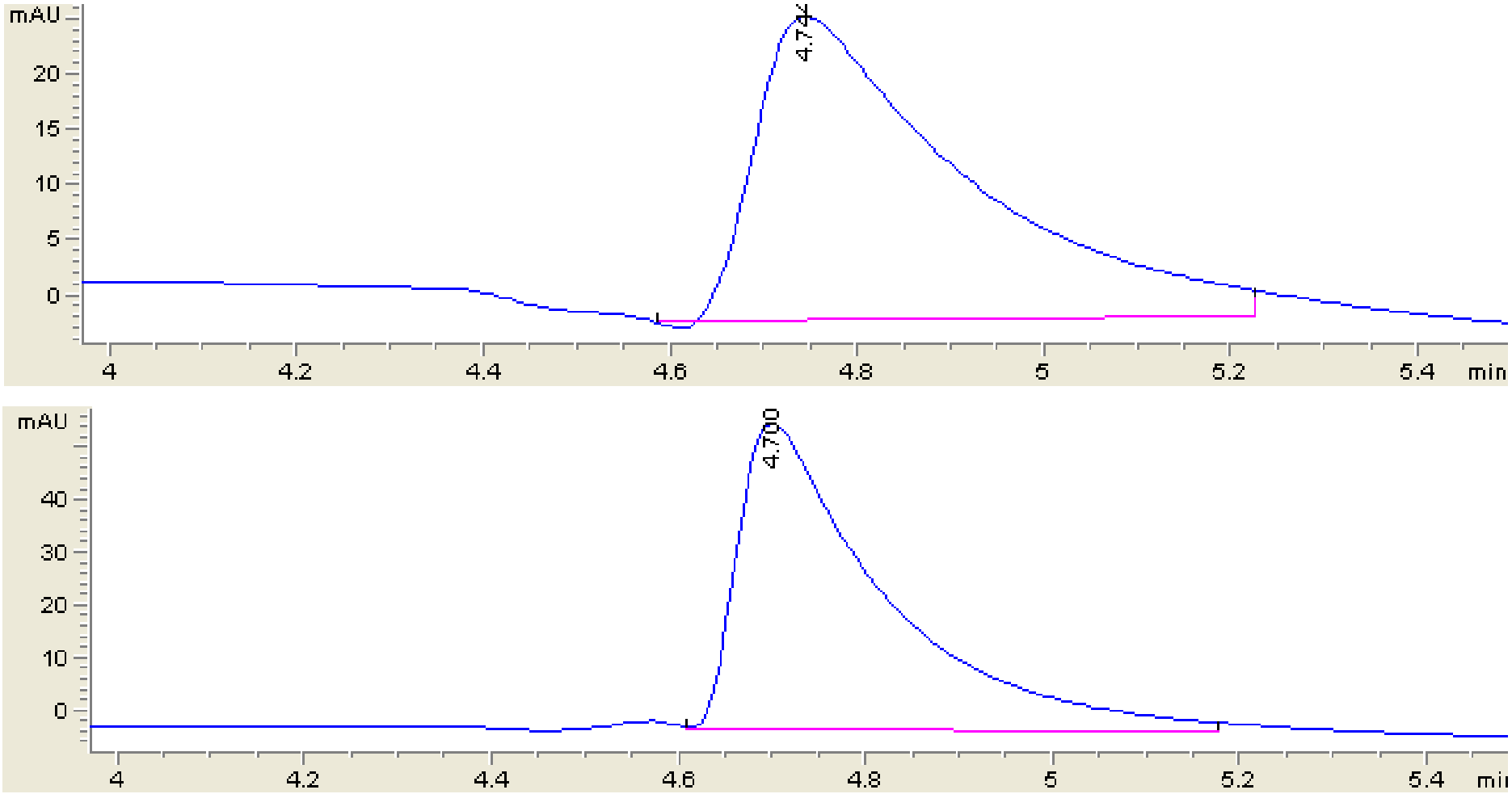


Figure 5.7: Excerpts of HPLC results for uninduced (Top) and induced (Bottom) samples of QT-WAHT66. There were elution peaks for uninduced and induced QT-TAHT66 samples, with corresponding Fab' concentration of 27.69mg/L and 31.75mg/L respectively. Full profiles of both HPLC results are available in Appendix I.

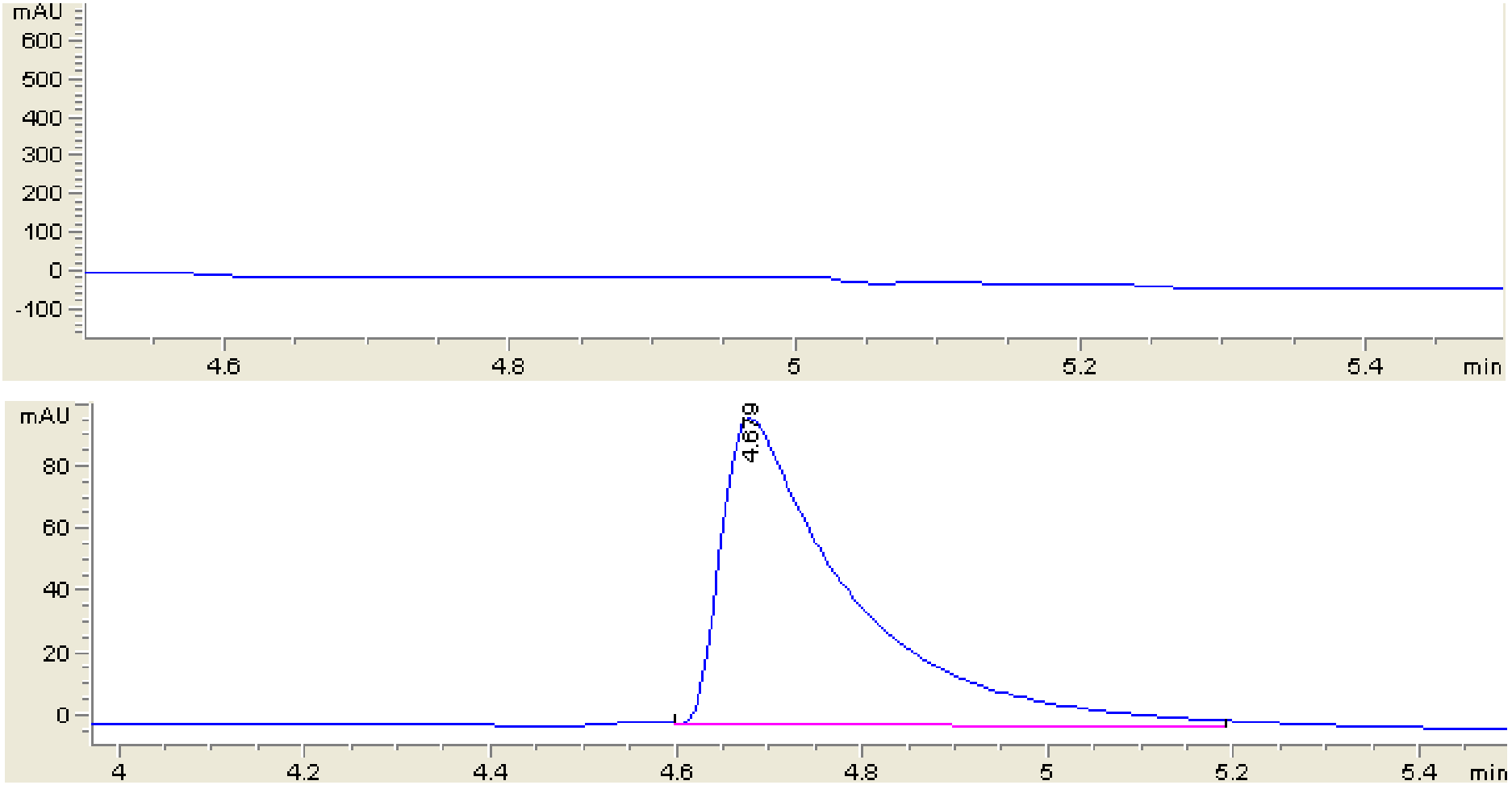


Figure 5.8: Excerpts of HPLC results for uninduced (Top) and induced (Bottom) samples of UCB. There was no elution peak for the uninduced UCB sample. When induced by IPTG, the UCB strain had an elution peak with a corresponding Fab' concentration of 46.37mg/L. Full profiles of both HPLC results are available in Appendix I.

The other theory that explains the presence of Fab' fragments in both uninduced and induced samples is that there was an issue with the plasmid design or transformation procedure. However, multiple other results strongly support the idea that it is not the latter. Firstly, it was unlikely to be a result of an error in the plasmid, since the resulting strain has resistance to three different antibiotics. It would have been highly coincidental for the wrong plasmid to be inserted and to possess the same three antibiotic resistances.

Secondly, while both uninduced and induced samples displayed peaks in the HPLC that indicate the production of Fab', the peaks for the uninduced samples were consistently lower than the induced samples. This behaviour reinforces the concept that the strain did behave as intended, with the problem being purely a deficiency in lac repressors in comparison to tacI promotors. Additionally, when comparing the transformed strains to wild type W3110 as depicted in Figure 4.3.2.B, observing the entire HPLC profile, the wild type did not produce any peaks in the HPLC even when induced by IPTG.

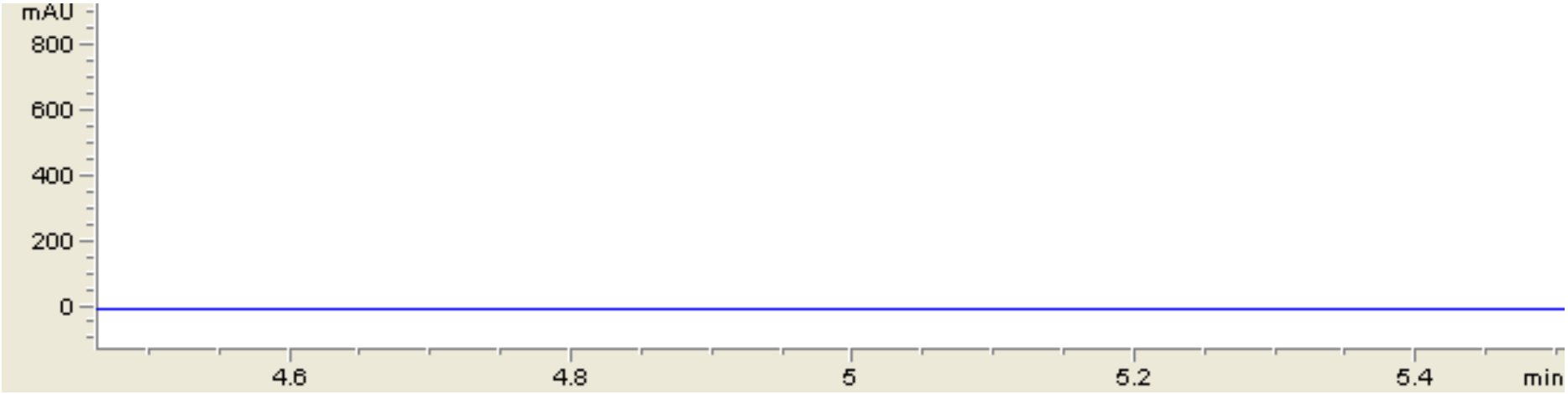


Figure 5.9: HPLC for wild type W3110 induced with IPTG. No peak could be observed after elution begins at minute 4, which indicated the lack of Fab' fragment presence.

5.4 Cultivation of *E. coli* Strain at Different Scales

The host strain W3110 has no intellectual property claims, thus the *E. coli* strain QT-WAHT66 was used moving forward in this project. A clone from a petri dish was selected at random and grown in comparison to the UCB strain for characterisation. This clone is named QT-EC.4.8.16.

5.4.1 50 mL Shake Flasks

QT-EC.4.8.16 and the UCB strain were both grown in 2XPY media, for 2 hours. After which, 1mL of the 2XPY broth was used to inoculate 50mL of SM6G media in 250mL shake flasks in triplicates and left to incubate overnight. After 10 hours, the growth was induced by IPTG, and 2mL samples were taken on a bihourly basis. 0.5mL was used for OD measurements, and Fab' was extracted from the remaining 1.5mL through sonication. The sample at 12 hours was taken immediately after the addition of IPTG.

As shown in Figure 5.10, QT-EC.4.8.16 reached a much lower OD when compared to the UCB strain. This was in part due to having both tetracycline and chloramphenicol antibiotics present in the medium, rather than tetracycline alone, inhibiting rapid growth. Additionally, it can also be a direct result of having two plasmids in the host strain, creating additional stress for the system, resulting in a lower achievable OD.

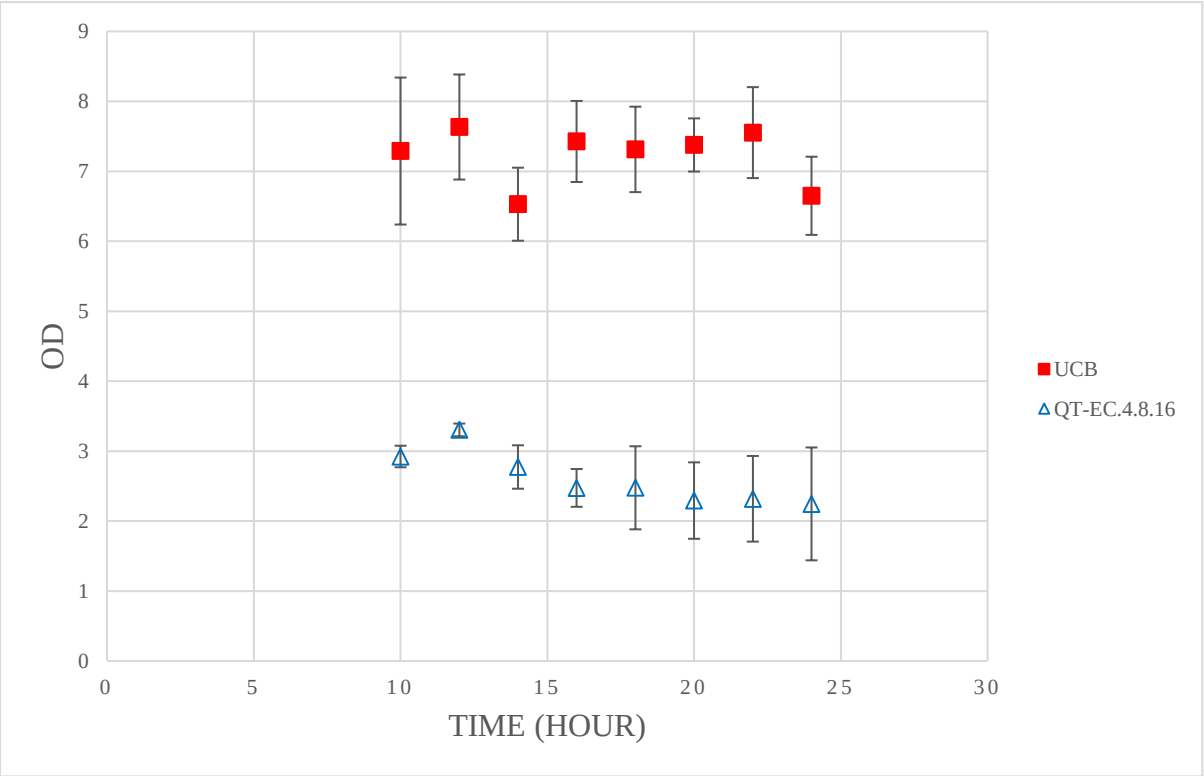


Figure 5.10: Growth profile of QT-EC.4.8.16 and UCB with error bars in 50mL of culture grown in shake flasks. Both strains reached a steady stationary phase after a period of 10 hours, with UCB reaching an OD between 7.0-8.0 and QT-EC.4.8.16 reaching a lower OD between 2.0-3.0.

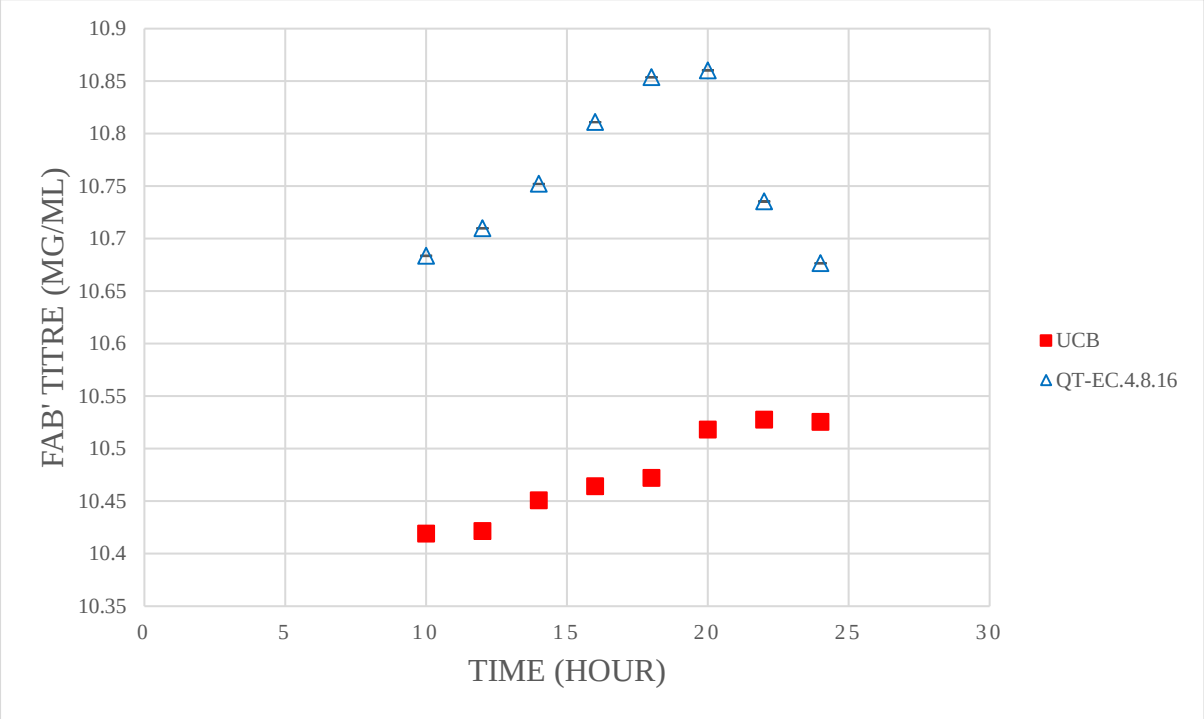


Figure 5.11: Fab' titre achieved in 50mL of culture grown in shake flasks for QT-EC.4.8.16 and UCB with their error bars. The errors values are very small and therefore not observable in the plot. QT-EC.4.8.16 achieved a slightly higher titre than the UCB strain.

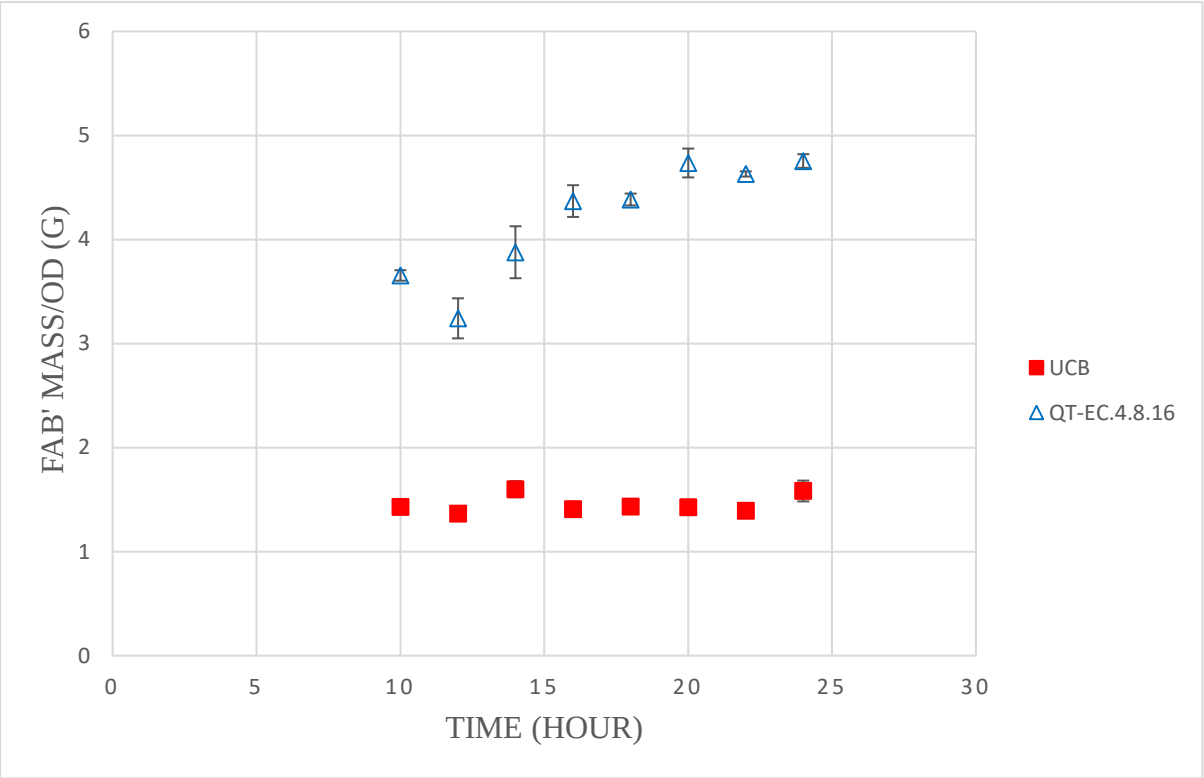


Figure 5.12: Fab' produced per OD for QT-EC.4.8.16 and UCB with error bars in 50mL of culture grown in shake flasks. The WAHT66 strain achieved a higher total Fab' mass than the UCB strain, however the UCB strain appeared to be more stable with lower errors during the stationary phase.

Although the OD achieved for QT-EC.4.8.16 was half of UCB, in the period both strains were monitored, QT-EC.4.8.16 produced a higher total Fab' weight than UCB as indicated in Figure 4.4.1B. While both strains achieved similar respectable Fab' weights, when considering the amount of Fab' produced per OD, shown in Figure 4.4.1C, QT-EC.4.8.16 appeared to have better yield.

It is interesting to note that the UCB strain produced a stable amount of Fab' per cell throughout the induction period. Whereas for QT-EC.4.8.16, the Fab' per cell increases to a maximum before taking a drop. The consistency of the UCB Fab' productivity could be attributed to the well-established nature of its host strain and optimised protocol and media composition. It is also likely that the variability in the performance of the QT-EC.4.8.16 was a relic of the interaction between the two plasmids transformed within the host.

5.4.2 800µL Cultivation in 96-Well Plate Using TECAN

On a small scale, high-throughput methods are used to analyse the performance of different clones of the same strain to select the optimal clone. The use of 96-well plates in the automated liquid handling system TECAN, allows for the automated monitoring of cell growth and productivity. Eight clones of QT-WAHT66 were monitored over a period of 34 hours to establish an understanding of how this strain performs in a small-scale environment.

1mL of glycerol stock was used to inoculate 10mL of LB media and left to incubate overnight. The OD was measured the following day, and a variable volume was used to inoculate fresh LB media to achieve a final OD of 0.2. The newly inoculated media was then distributed amongst the wells in 800µL portions. For ease of comparison, IPTG was introduced at the beginning of the incubation period. This should allow the tac promoter to act completely uninhibitedly, producing Fab' constantly.

The EVOWare script for the TECAN took 50µL OD samples and 500µL Fab' samples from a sacrificial well for each clone over the 36 hours period. OD samples were placed in a 96-well flat bottom plate, which was transferred to the plate reader for OD measurements. Four-fold and, if required, tenfold dilutions were made if the OD measurement exceeds 1. Fab' samples were placed in a 1.2mL 96-well plate. This plate was transferred to the attached centrifuge and Fab' was extracted from the periplasm of the resulting pellet using osmotic shock. Sonication could not be used in this configuration, as the equipment required was not available. After the Fab' was extracted, the samples were transferred to a HPLC for analysis.

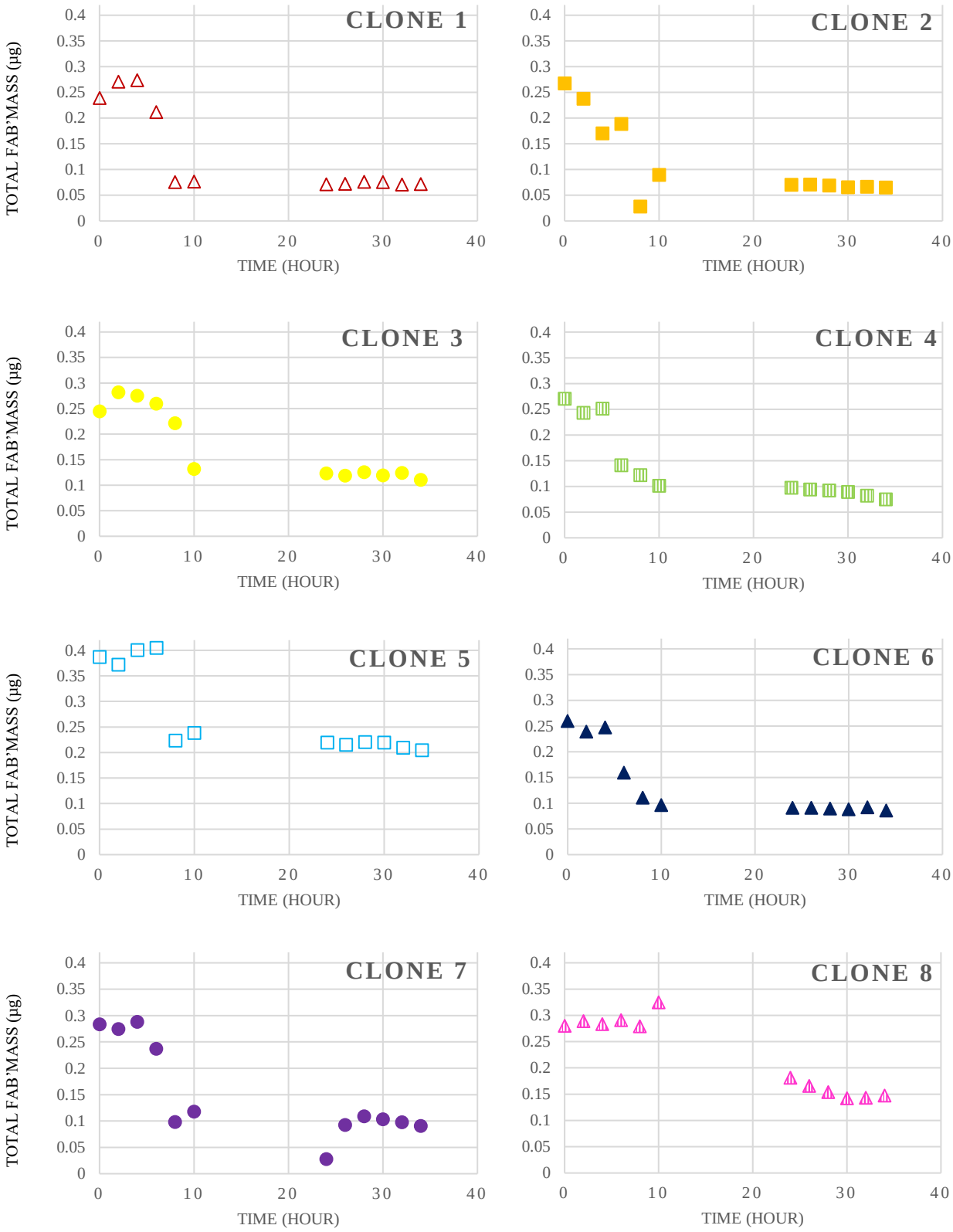


Figure 5.13: Individual plots of total Fab' mass produced in 800µL wells for eight different QT-WAHT66 *E. coli* clones grown at a small-scale cultivation in the TECAN.

Selecting three of the best performers, Clone 5, Clone 7 and Clone 8, a more comprehensive run was undertaken in which multiple samples were taken per time point. This provided a clear depiction of how these three clones performed. Using the same method, the 96 wells were shared equally between the three clones, and 4 sets of samples were taken over a period of 28 hours.

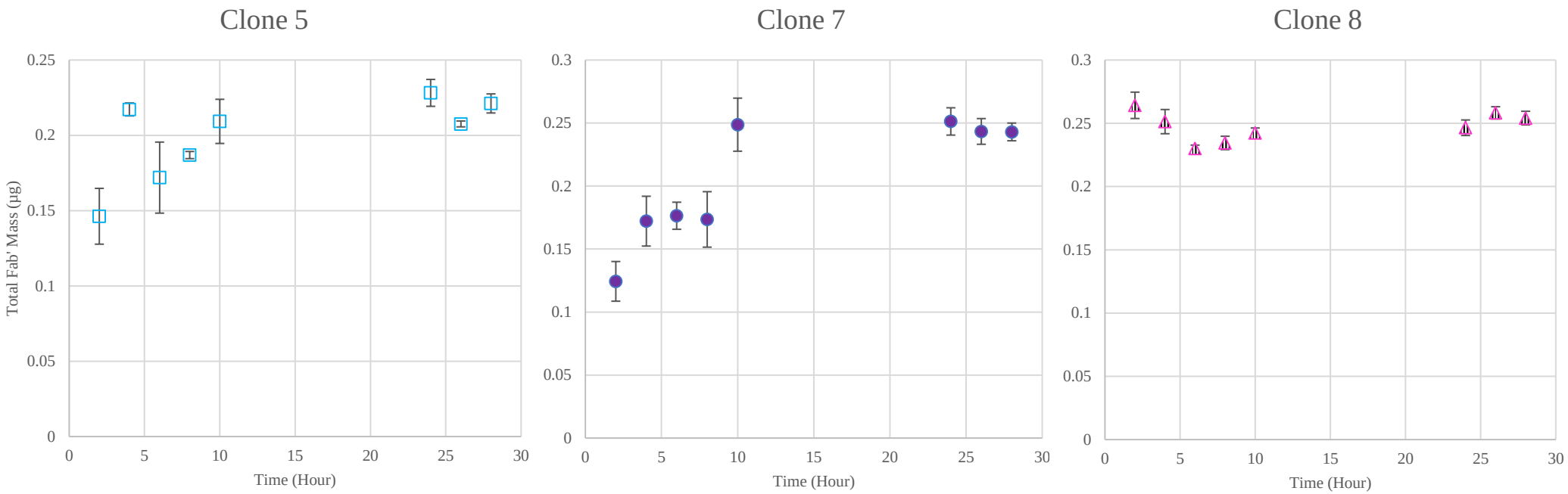


Figure 5.14: Individual plots of Fab' mass per OD in 800µL wells for QT-WAHT66 *E. coli* Clone 5, Clone 7 and Clone 8 grown at a small-scale cultivation in the TECAN.

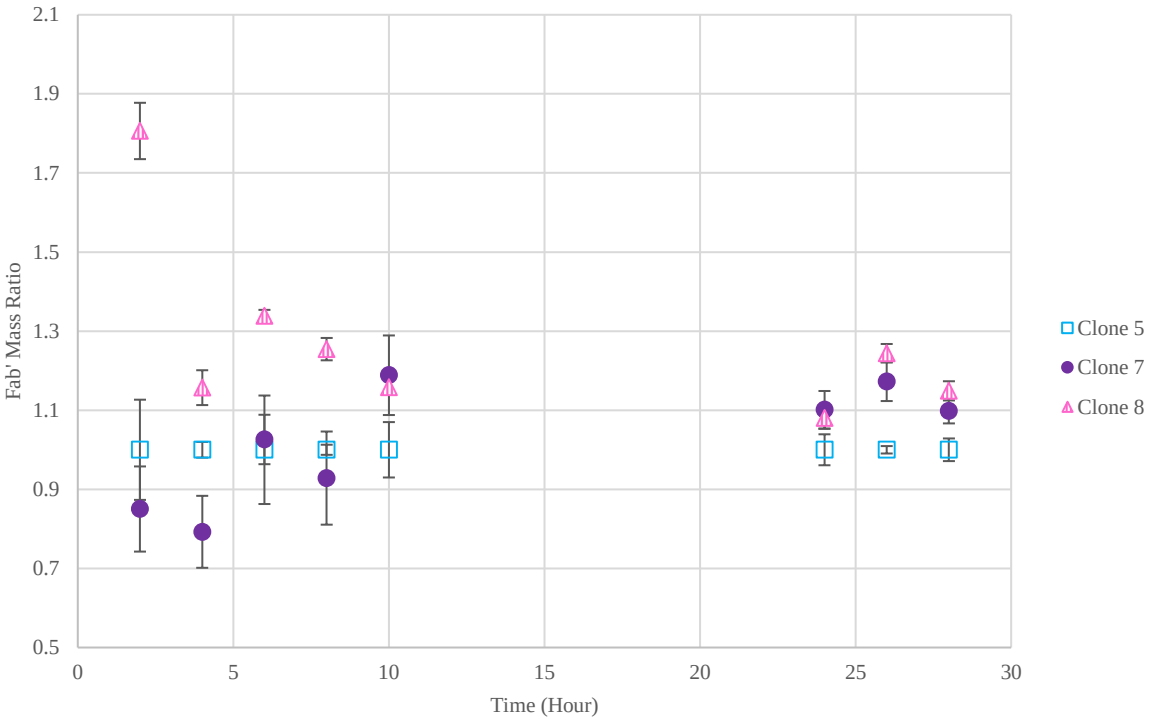


Figure 5.15: Ratio of Fab' Mass for all three clones normalised to the lower of the three, Clone 5, in this instance.

The results indicated that Clone 8 had the highest productivity per cell, with Clone 7 being the viable second choice. The larger error bars at the early data points for Clone 5 and Clone 8 correlated with the more erratic behaviour of its productivity profile.

5.4.3 200mL DASBox Cultivation

While the TECAN is a useful tool for high-throughput analysis, industrial fermentations are rarely conducted at such a small scale. For the conclusions drawn from the small-scale studies to hold validity, the results must hold true when scaled-up. Therefore, the same three clones were grown in DASGIPs fermenters and their OD and productivity monitored. If the clones' performance in the TECAN was reflected in the DASGIPs, it is an indication that the conclusions drawn from high-throughput small-scale studies can be extrapolated to larger scale cultivation vessels.

25mL of LB media was inoculated and grown until the OD was between 3-4. 20mL of this broth was used to seed 180mL of LB media in each DASGIPs, giving a total volume of 200mL. To use the same conditions as in the TECAN, IPTG was added at inoculation. The temperature, dissolved oxygen level and pH of the media was regulated by the DASBOX, and the cultivation growth was monitored over a two-day period. Samples were taken for OD and Fab' measurements.

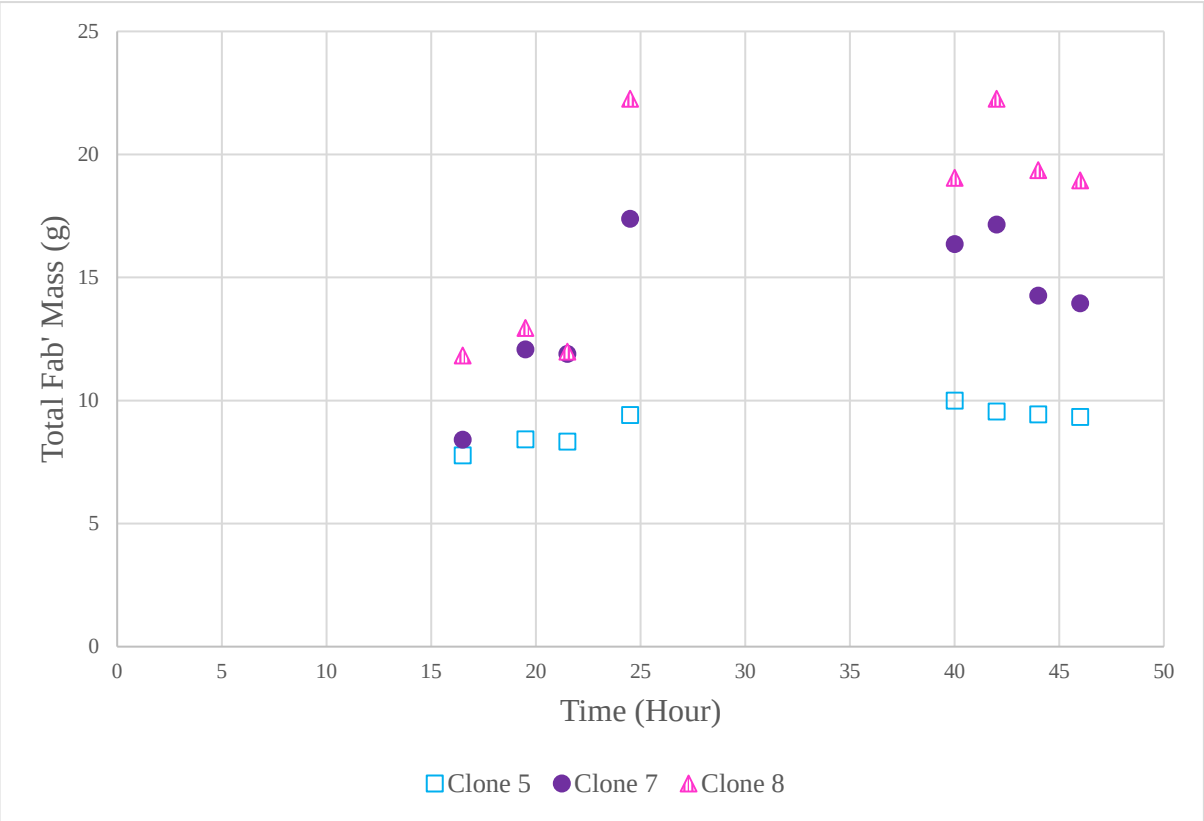


Figure 5.16: Total Fab' mass for Clone 5, Clone 7 and Clone 8 of the QT-WAHT66 *E. coli* strain cultured in 200mL DASGIP vessels. Each data point is the mean of 3 technical repeats. Error bars were included but were too small to be seen.

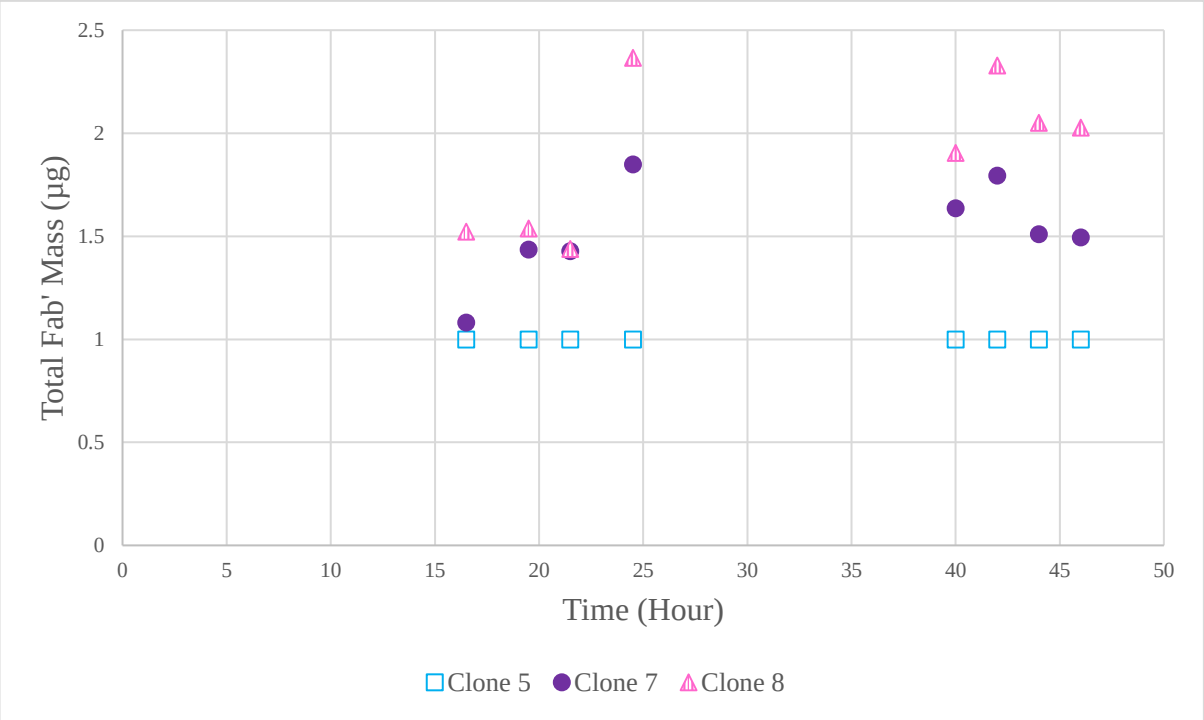


Figure 5.17: Ratio of Fab' Mass for all three clones normalised to the lower of the three, Clone 5, in this instance.

Production at large-scale prioritises the titre achieved, therefore, the evaluation of the clones should be based solely on the total Fab' weight yield. When comparing Figure 5.15 to Figure 5.14 in the previous section, it can be noted that the three clones performed the same in both large and small scale relative to each other; With Clone 8 reaching the highest total Fab' weight and Clone 5 severely underperforming relative to the other two.

This reflection suggests that the optimisation work performed using the TECAN can be extrapolated and applied to larger cultivation vessels. The different clones' relative performances remain the same when scaled-up.

5.5 Chapter Conclusion

The Anti-hepatitis B Fab' sequence was optimised for expression in bacterial cells. The genetic sequence expressing the Fab' was written and cloned into the low copy number plasmid, pACYC184. The AHFEC-H2T and pMMB66EH were both successfully transformed into W3110 *E. coli* cells creating the QT-WAHT66 strain, inducible by IPTG. This strain is capable of producing the intended Fab' at 800µL, 50mL and 200mL scale. It was shown using three clones that the relative ranking of the clones remains the same from 800µL to 200mL.

Chapter 6: Designing and Establishing Fab' Expression *P. Pastoris*

6.1 Modifying the Anti-HBS mAb Sequence for Expression as Fab' in *P. pastoris*

Yeast cells are an effective host system and the intermediary between *E. coli* and CHO cells. Since it is both microbial and eukaryotic, it retains the benefits of both systems. They are robust with relatively short doubling time and possess the ability to produce proteins in its correct tertiary form. Unlike *E. coli*, where a refolding process might be necessary, yeast cells have evolved protein folding pathways that are more suitable for heterologous proteins and, overall, can produce proteins identical to naturally occurring ones (Frenzel, et al., 2013). As a result, little to no modifications would be required to adapt the sequence for yeast host cells. The DNA sequence was optimised for expression in *P. pastoris* by using the same codon optimisation software as mentioned in the previous chapter.

6.1.1 Considerations Regarding Glycosylation in *P. pastoris* Gene Design

A consideration is that yeast forms both N- and O-glycosylation sites slightly differently from mammalians cells, and therefore the target protein might function differently (Verma, et al., 1998). While this might pose as a problem for other heterologous protein expression, the lack of glycosylation sites in the Anti-HBS Fab' rendered this issue insignificant.

Human post-translation modifications are not fully optimal in yeast cells, and therefore not all proteins can be produced successfully in *P. pastoris* (Kunert, et al., 2008). The tendency for hyperglycosylation of heterologous proteins is a common issue. This causes glycosylation to occur at unintended points, affecting the structure of the protein.

Post-translational performance is largely dependent on the complexity and structure of the intended protein product, with less complex proteins being expressed with greater success. *P. pastoris* also glycosylates more similarly to mammalian cells and exhibits less hyperglycosylation than other yeast hosts (Kunert, et al., 2008). Since the Fab' was produced in the far more fastidious prokaryotic hosts, the more sophisticated folding pathway of the *P. pastoris* host should not struggle with folding the tertiary structure of the same Fab'.

6.2 Construction of the Expression Cassette for *P. pastoris*

6.2.1 Intellectual Property Free pJ90x-15 Plasmid for Strain Construction

The plasmid used for yeast expression of the Anti-HBS Fab' is pJ902-15. This plasmid has a pUC, high copy number, bacterial origin. This feature will prove to be useful when trying to produce and purify enough DNA for a transfection into *P. pastoris*. It has a zeocin resistance gene and has a stuffer protein, Rudolph Red Fluorescent Protein. The fluorescent protein is flanked by varies unique restriction sites, using these the sequence for stuffer protein will be removed and replaced with the Fab'. The plasmid map is shown in Figure 6.2.1A.

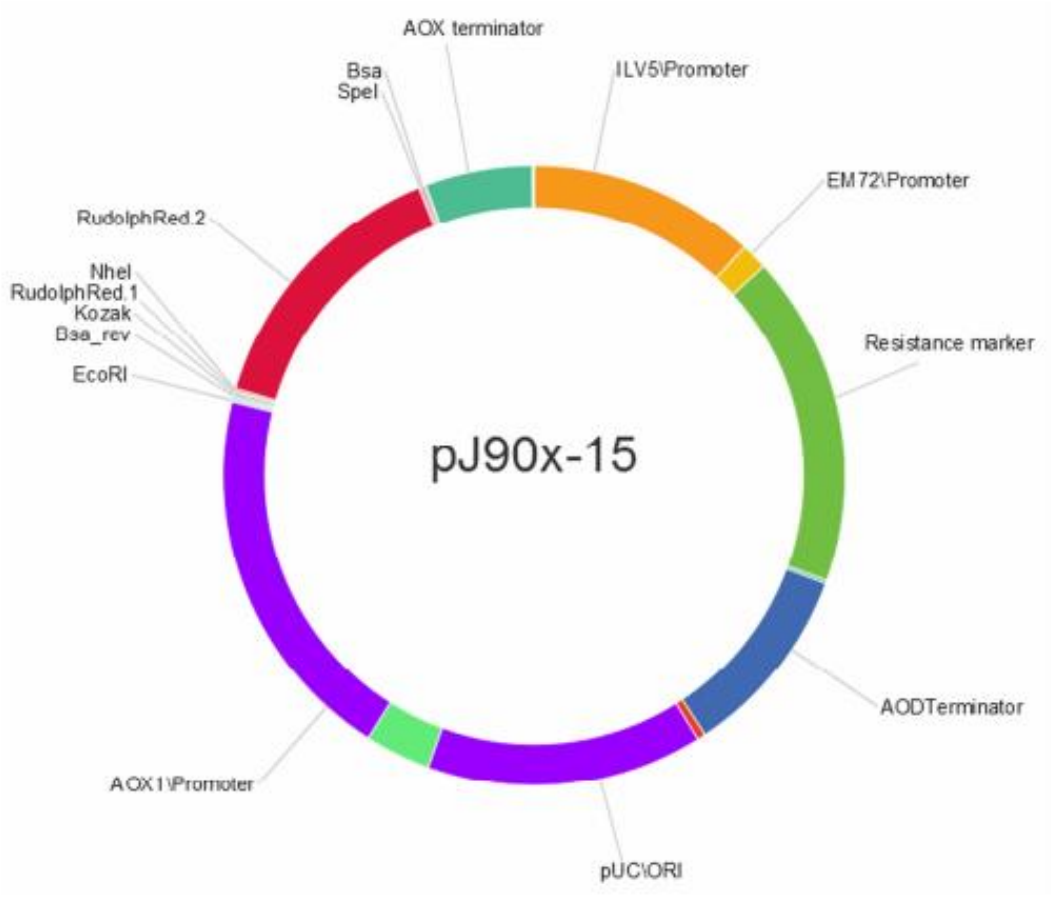


Figure 6.1: The plasmid map of pJ902-15, showing the restriction sites and the location of the origin of replications and resistance markers.

6.2.2 Promoter and Secretion Signal

The backbone plasmid already possesses a AOXI promoter and AOX terminator. Since the promoter and terminator correspond with each other, the insert must use the same promoter and terminator for both the heavy and the light chain.

P. pastoris is known for high yields, and the resulting stress from proteins accumulated within the host cells would prompt cell apoptosis (Yu, et al., 2015). Thus, secretion of the product is paramount to preserving healthy cells and thereby maintaining a stable product yield. An α-factor secretion signal is therefore upstream of both heavy and light chain.

6.2.3 Fab' Expression Cassette for *P. pastoris*

As mentioned previously, the backbone plasmid already has a set of AOXI promotor and terminator. Therefore, the insert design does not need to include the AOXI promotor for the heavy chain, and the AOXI terminator for the light chain. The insert is flanked by two unique restriction sites NheI and SpeI.

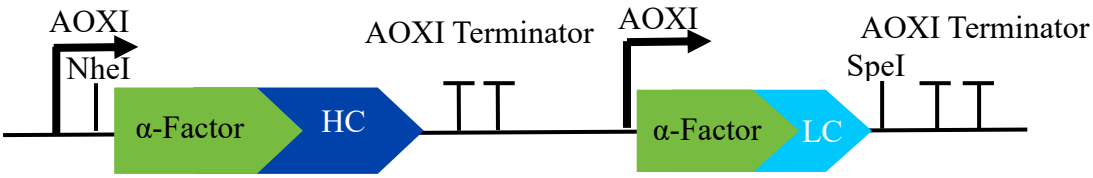


Figure 6.2: Design cassette for the expression of the anti-hepatitis B Fab' fragment in *P. pastoris*. An α-factor secretion signal is placed upstream of both the heavy and light chains. Both sequences are promoted and terminated using an AOXI promoter and terminator. The insert is placed on a pJ902-15 backbone.

6.3 Generating the *P. pastoris* Strain for Fab' Expression

The insert depicted in Figure 6.2.4A is synthesized in collaboration with Eurogentec. The insert is then cloned into pJ902-15, replacing the Rudolph Red Fluorescent Protein, via the NheI and SpeI unique restriction sites. The synthesised plasmid is transformed into Top10 *E. coli* strain.

6.3.1 Transformation Strategy A

The Top10 strain containing the plasmid is grown in low salt LB and plasmid is purified from the resulting overnight media. The plasmid is linearized using SacI restriction site as suggested by the Invitrogen Manual. Linearization with this restriction site favours the isolation of His^+ Mut^+ recombinants in GS115. 15µg of linearized DNA is used to transform into electrocompetent GS115 using electroporation and selected for successful transformation using zeocin antibiotics resistance.

The seeming successfully transformed colonies were picked and grown in 10mL of media then induced with methanol. The resulting HPLC analysis revealed that no Fab' product was produced for the wildtype, uninduced and induced samples. The transformed samples are able to grow in the presence of zeocin, and therefore the lack of Fab' being produced is unlikely a result of a failed transformation attempt.

Investigating the cause of this revealed that the SacI restriction site is present within the AOXI promotor. Unfortunately, because this promotor is used twice within the insert, when linearizing with SacI the plasmid was cleaved into two sections. As a result, the heavy chain was isolated and only the light chain of the Fab' was transformed into GS115 with the rest of the pJ902-15 plasmid. This caused the resulting colonies to possess the zeocin resistance yet is unable to produce a complete Fab'.

6.3.2 Transformation Strategy B

An alternate linearization site had to be used for the transformation to include both chains in the insert. All the suggested linearization sites are present within the insert and therefore not unique. The insert and plasmid sequences are searched for a unique restriction site. However, it cannot be within any of the AOXI promoters, terminators, Anti-HBS insert, and the zeocin resistance gene. SwaI is a unique restriction site located between the AOXI promotor and the pUC origin of replication. This site was used to linearize the plasmid for transformation via electroporation. As this is not a standard linearization site, the transformation cannot guarantee any dominate phenotypes.

6.3.3 Measuring Fab' in Induced Transformed *P. pastoris* Cells

The second transformation attempt was successful, generating colonies that were resistant to zeocin. Three colonies were picked and grown in 10 mL of media. Due to the α factor secretion signal, the produced Fab' would be secreted into the media. After an overnight incubation at 30 ^{o}C and 200 RPM, the broth was centrifuged, and a sample of the supernatant was placed in the HPLC for analysis. Three different colonies grown in BMMY produced 16-18 mg/L, whereas the same colonies produced no Fab' when grown in BMGY. Additionally, wild type GS115 also produced no peaks in the HPLC. The GS115 strain successfully transformed with the designed plasmid is named QT-GAHT.

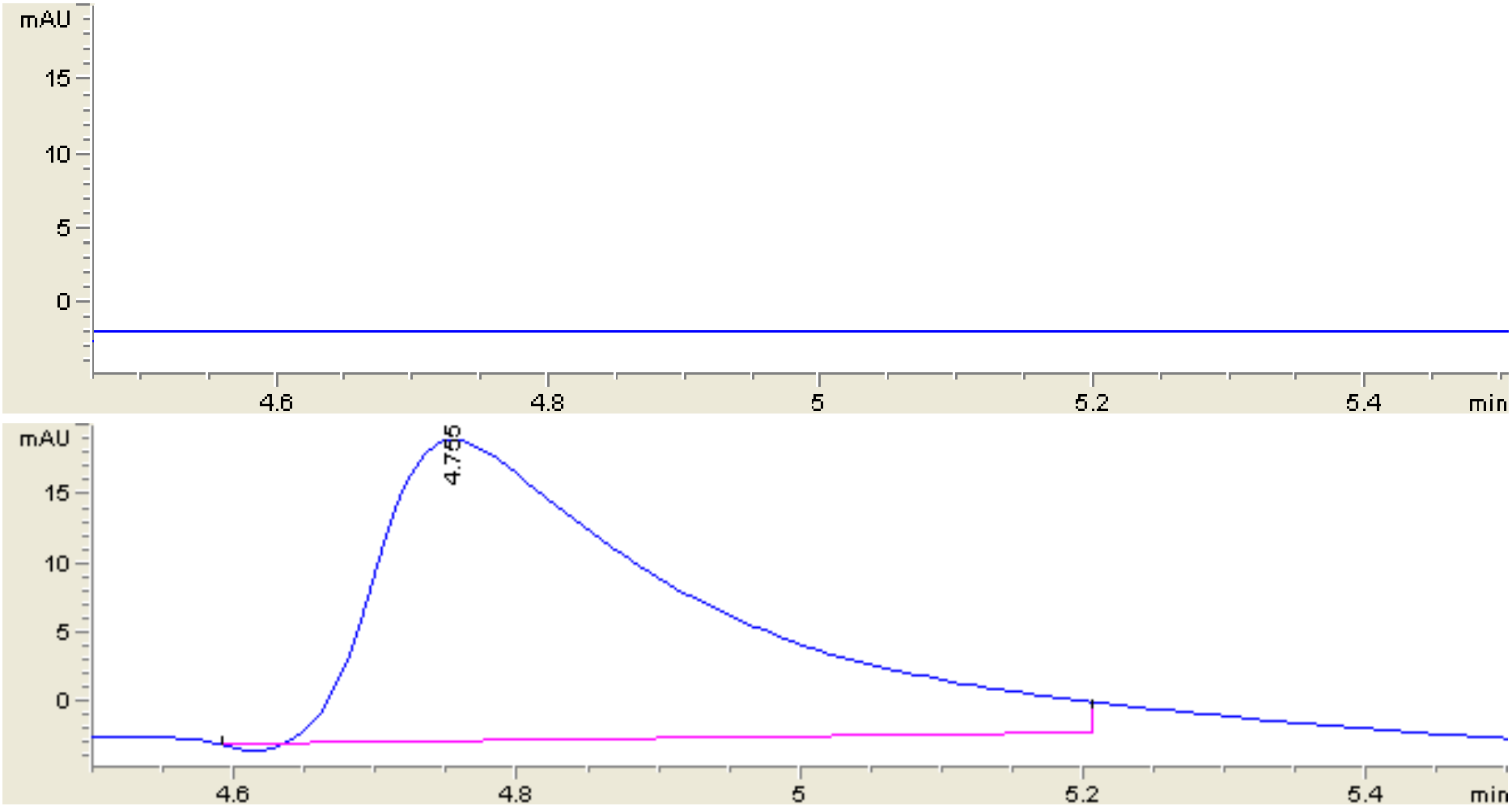


Figure 6.3: HPLC results for *P. pastoris* cells. Wild type GS115 (Top) produces no peak when induced with methanol. QT-GAHT (Bottom) produced an elution peaks when induced with methanol, with a corresponding Fab' concentrations of 17.9 mg/L. Full profiles of the HPLC results can be seen in Appendix II.

6.4 Cultivation of *P. pastoris* Strain at Different Scales

6.4.1 50 mL Shake Flasks *P. pastoris* Culture

Growth and Fab' production of a Mut^+ *P. pastoris* strain in complex glycerol media and complex methanol media is monitored. This clone was selected randomly from a petri dish and will be referred to as QT-PP.28.9.17. As a negative control, wild type GS115 is grown in complex glycerol media.

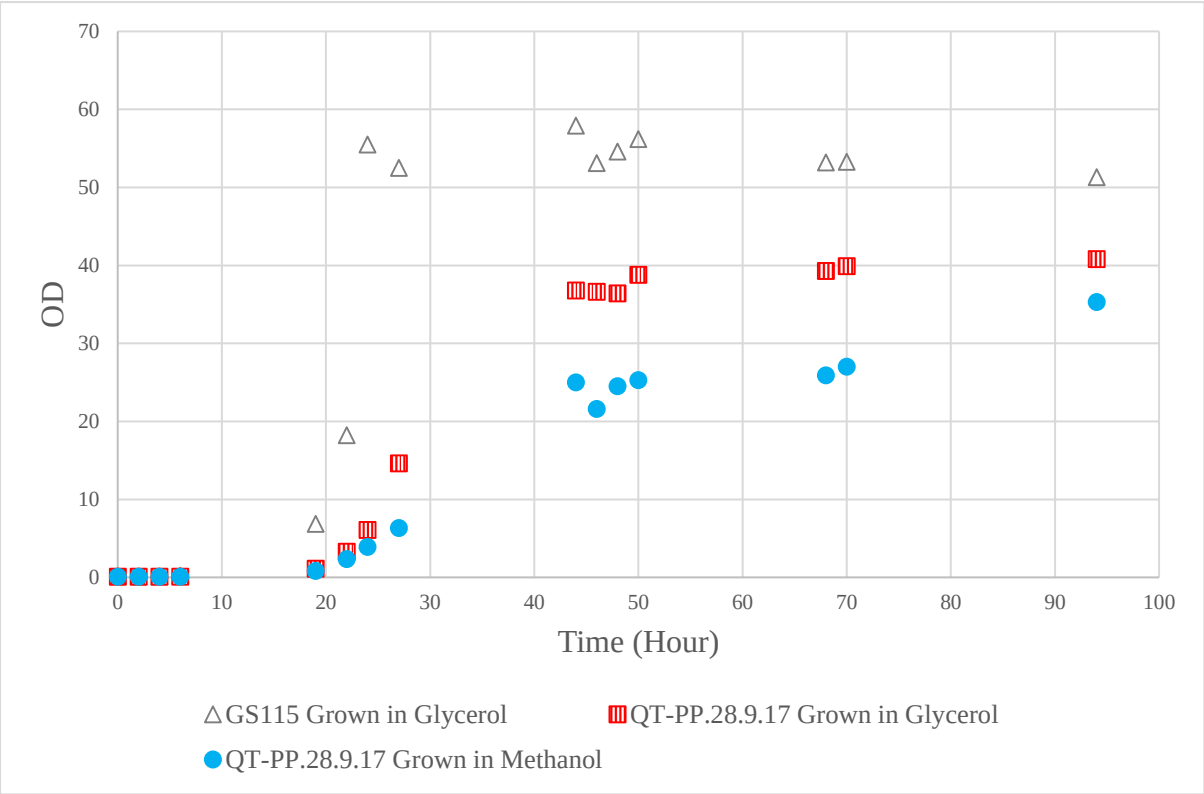


Figure 6.4: Growth profile of the QT-PP.28.9.17 strain in methanol and glycerol, and wild type GS115 in glycerol in 50mL shake flask culture. Error bars were plotted but unperceivable.

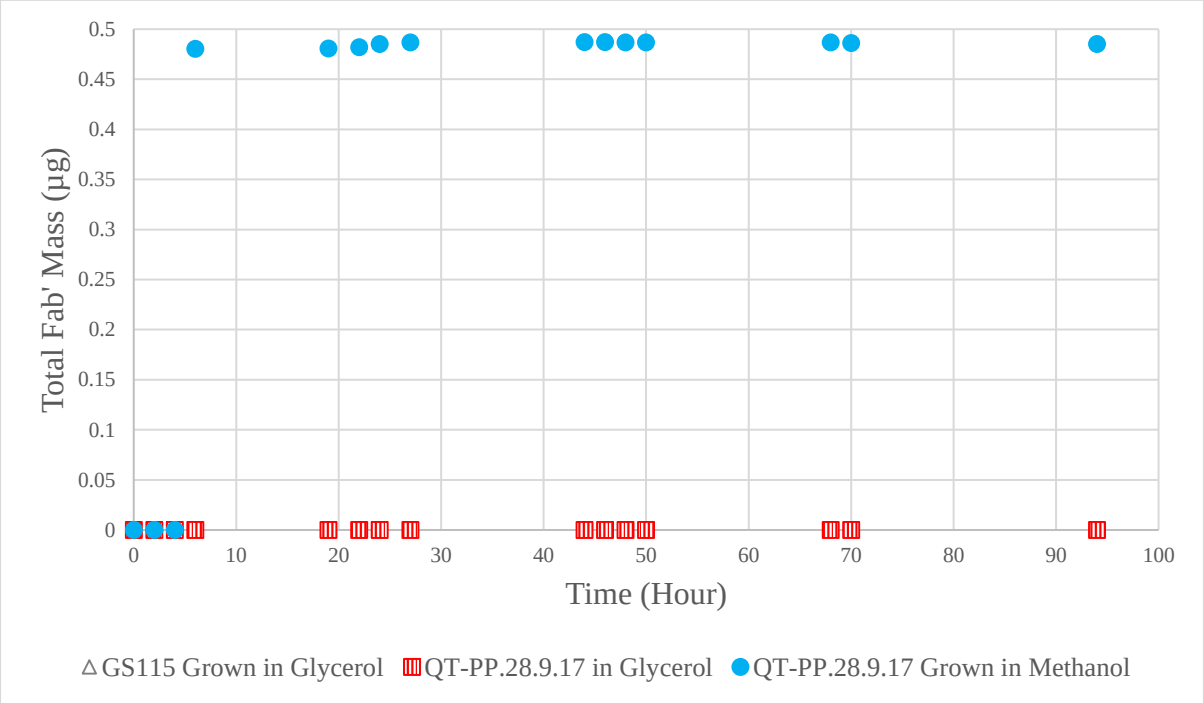


Figure 6.5: Total Fab' mass produced for QT-PP.28.9.17 strain in methanol and glycerol, and wild type GS115 in glycerol as negative control in 50mL shake flask culture. QT-PP.28.9.17 grown in glycerol does not produce any Fab' fragments.

The wild type GS115 strain grew more rapidly and to a higher OD compared to the QT-PP.28.9.17 clone. With glycerol as a carbon source, QT-PP.28.9.17 achieves a higher cell density, however, as can be seen in Figure 6.4.1B, it does not generate any Fab' fragments. The AOXI is working to its full effect, effectively inhibiting the Fab' production until induced by changing the carbon source to methanol.

6.4.2 800µL *P. pastoris* Cultivation in 98-Well Plate Using TECAN

Observing from the shake flask growth and Fab' production, productivity reaches a plateau around 24 hours after inoculation, and holds steady until it slowly decreases starting at 80 hours. Evaporation of the growth media is a concern when growing *P. pastoris* cells at the small scale of 800µL over a 4-day period. From experience with growing *E. coli* in the TECAN platform, media does evaporate at considerably even after 24 hours. Additionally, since methanol would be used as a carbon source for the media, its evaporation rate is higher than other alternatives. Hence, an evaporation study was conducted to determine the rate of evaporation of the methanol media in 800µL wells.

To replicate the same growth conditions, the TECAN was scripted to perform the exact same tasks and the 96-deep well plate was inoculated exactly as it would be for normal small-scale cell growth. After a 24-hour period, the plate was removed, and the remaining media volume is measured. Selecting 20 randomly distributed wells, the remaining media averaged to 550µL. Approximately 31% of the methanol media has evaporated over 1 day. This would significantly affect the results of cell growth, especially if conducted over 96 hours.

A media top-up was added into the TECAN script for *P. pastoris*. 250µL of methanol media is added to each well every 24 hours before sampling. The added volume returns the total volume of each well back to a working 800µL. While this keeps the working volume relatively constant

throughout the 4-day incubation period, it does introduce other considerations. Most importantly, the addition of fresh media provides additional nutrients which would help the cells maintain its cell density. Hence when compared to growth of *E. coli* in TECAN, the *P. pastoris* cells would appear to have a longer stationary growth phase. However, this would translate better into large scale fermentation, as fed batch are more common in larger vessels.

Glycerol stock was prepared for 8 random clones of QT-GAHT picked from a petri dish. 1mL of glycerol stock was used to inoculate 10mL of YPD media and left to incubate overnight. The OD was measured the following day, and a calculated volume was used to inoculate BMMY media to achieve a final OD of 0.2. The newly inoculated media was then distributed amongst the wells in 800 µL portions. To imitate the same effect as in *E. coli*, methanol media was used from the beginning of cultivation to allow for the continual production of Fab'.

The EVOWare script for the TECAN takes 50µL OD samples and 500µL Fab' samples from a sacrificial well for each clone over a 48 hours period. OD samples were placed in a 96-well flat bottom plate, which were transferred to the plate reader for OD measurements. tenfold and, if required, 100-fold dilutions were made if the OD measurement exceeds 1. Fab' samples were placed in a 1.2mL 96-well plate. This plate was transferred to the attached centrifuge and Fab' was isolated from the *P. pastoris* cells. The supernatant was then transferred to the HPLC for analysis.

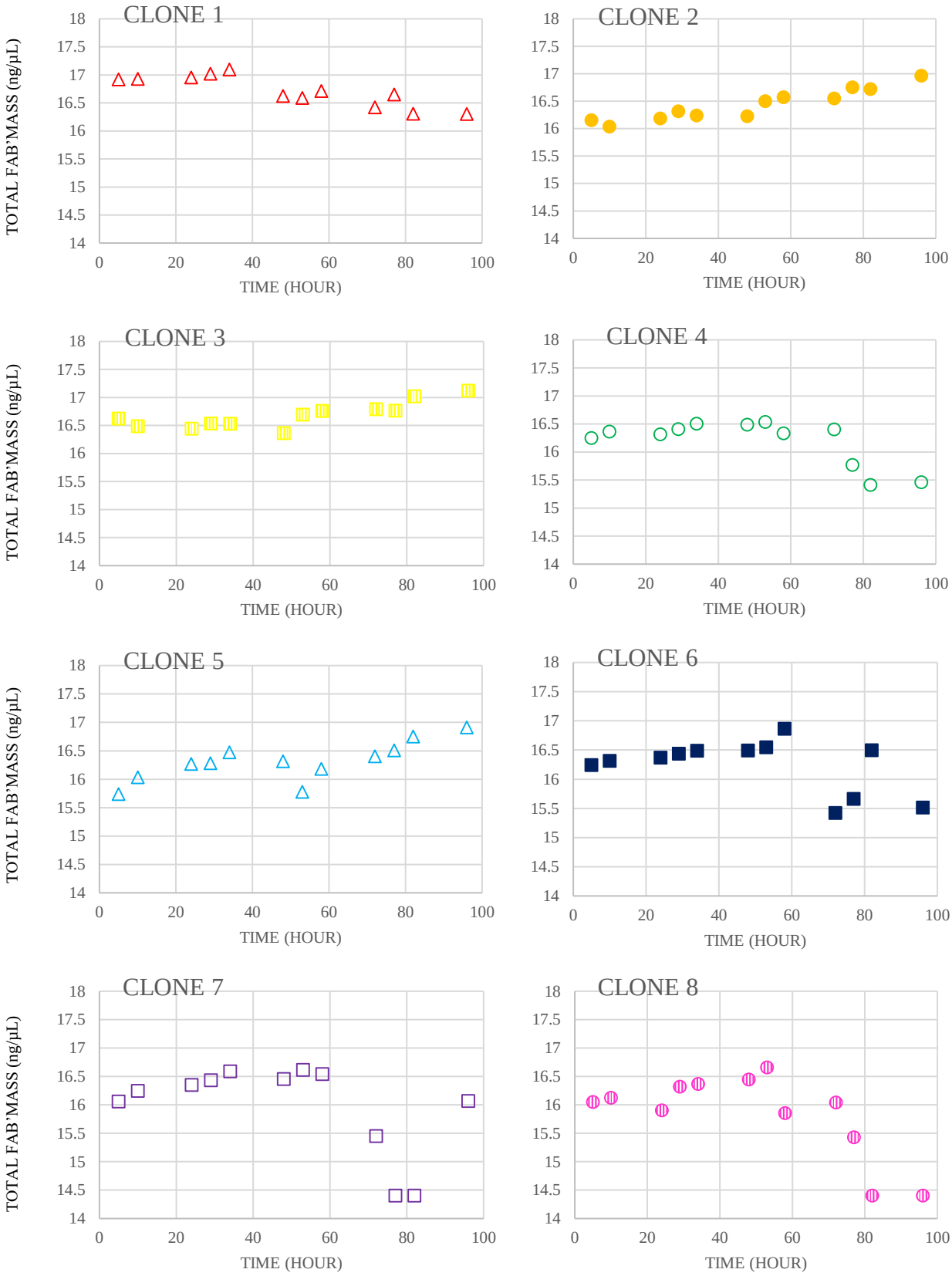


Figure 6.6: Individual plots of total Fab' mass produced in 800µL wells for eight different QT-GAHT *P. pastoris* clones grown at a small-scale cultivation in the TECAN.

Figure 6.6 shows Clone 2, Clone 3, and Clone 5 are selected for their high productivity. Clone 1 was discounted because the total Fab' weight decreased over the cultivation period. Although Clone 6 reached a higher total Fab' weight than the others at 55 hours, its behaviour thereafter is too erratic. The three selected clones are then grown in quadruplicates in the same manner. 8 sets of quadruplicate samples will be taken over the course of 90 hours.

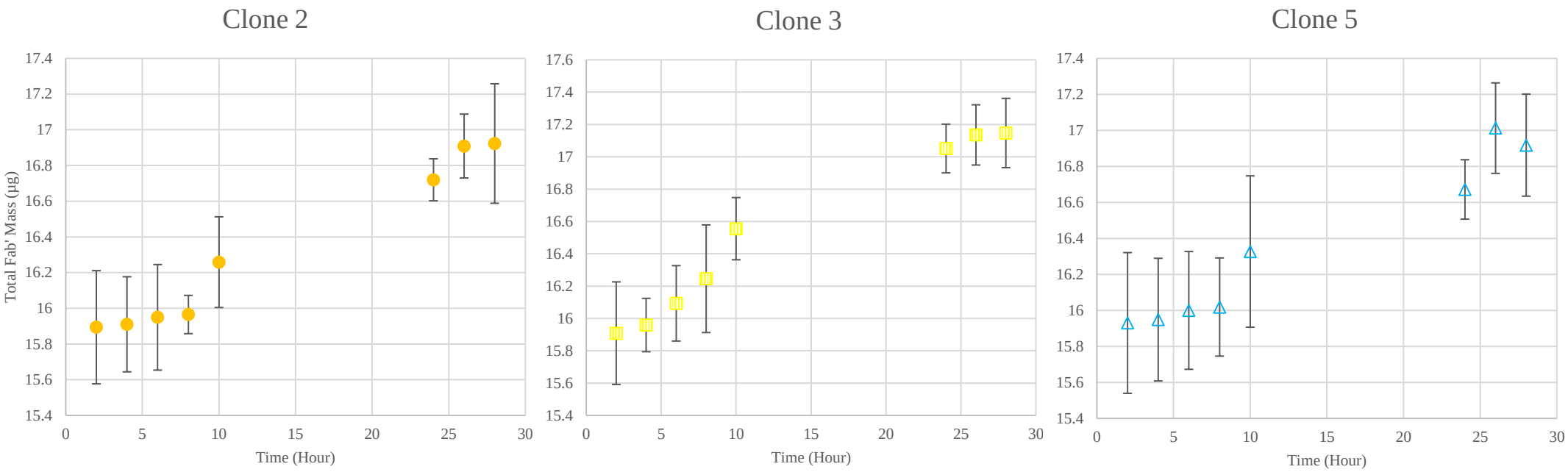


Figure 6.7: Individual plots of specific yield for QT-GAHT *P. pastoris* Clone 2, Clone 3 and Clone 5 in 800µL well, grown at a small-scale cultivation in a TECAN.

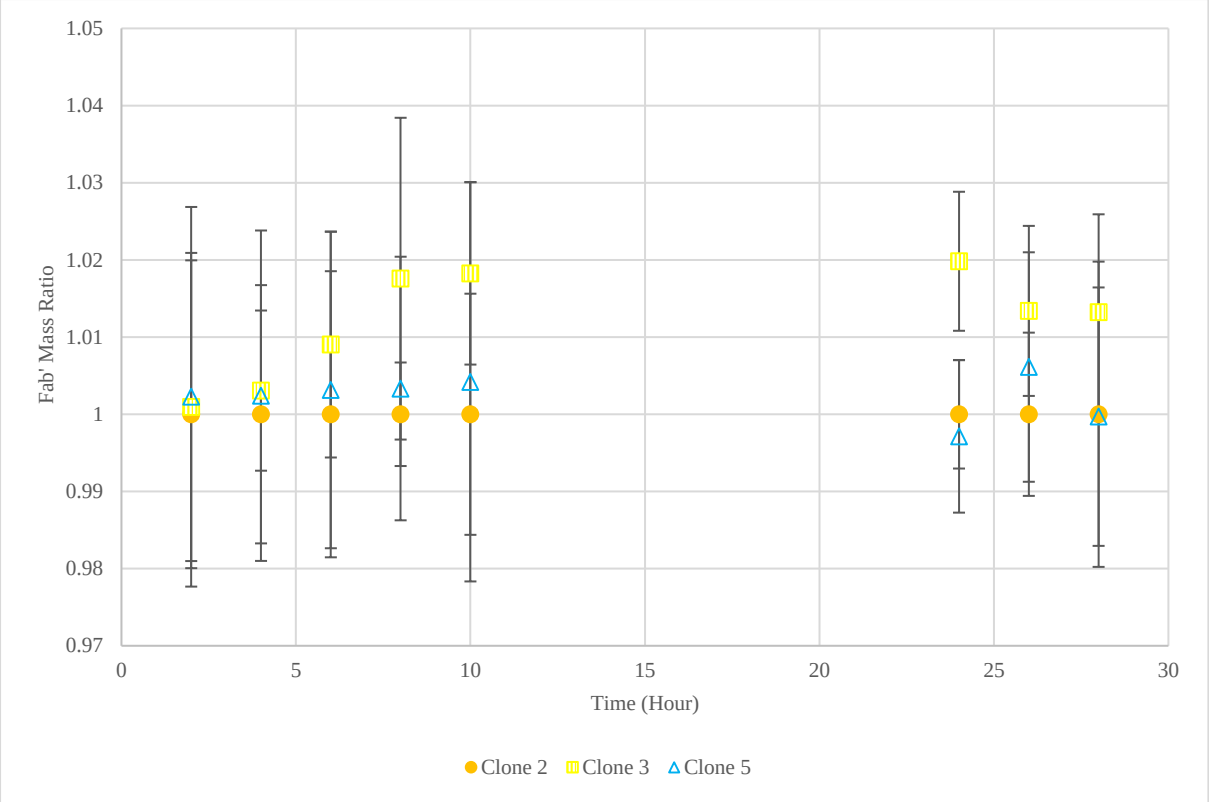


Figure 6.8: Ratio of Fab' Mass for all three clones normalised to the lower of the three, Clone 5, in this instance.

Although it appears the amount of Fab' produced decreases over the cultivation period, this is incorrect. Looking at Figure 5.4.2G, when the total Fab' mass is plotted, the total Fab' mass increases over the course of the cultivation period. The low OD at the beginning of the cultivation period would artificially increase value of relative Fab' mass. Clone 3 achieve a higher stable Fab' mass compared to Clone 2 and Clone 5.

6.4.3 200mL DASBox Cultivation for *P. pastoris*

25mL of YPD media is inoculated and grown until the OD is between 3-4. 20mL of this broth is used to seed 180mL of BMMY media in each DASGIPs, giving a total volume of 200 mL. The temperature, dissolved oxygen level and pH of the media is regulated by the DASBOX, and the cultivation growth is monitored over a four-day period. Samples are taken for OD and Fab' measurements. Throughout the cultivation period, the volume of the vessels was observed. Possibly due to being in an enclosed vessel, the media volume did not have the same significant evaporation as experienced in the smaller scale.

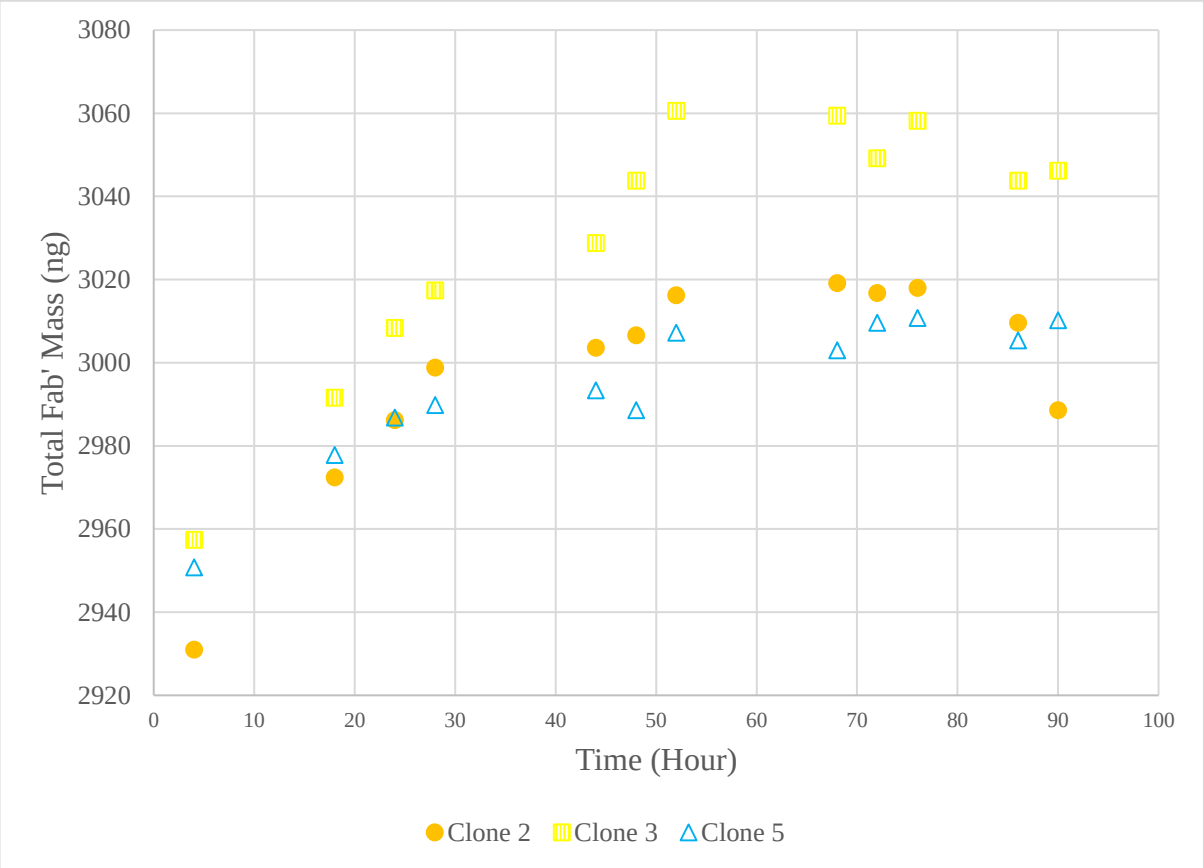


Figure 6.9: Total Fab' mass for Clone 2, Clone 3 and Clone 5 of the QT-GAHT strain cultured in 200mL DASGIP vessels. Error bars were included but were too small to be seen.

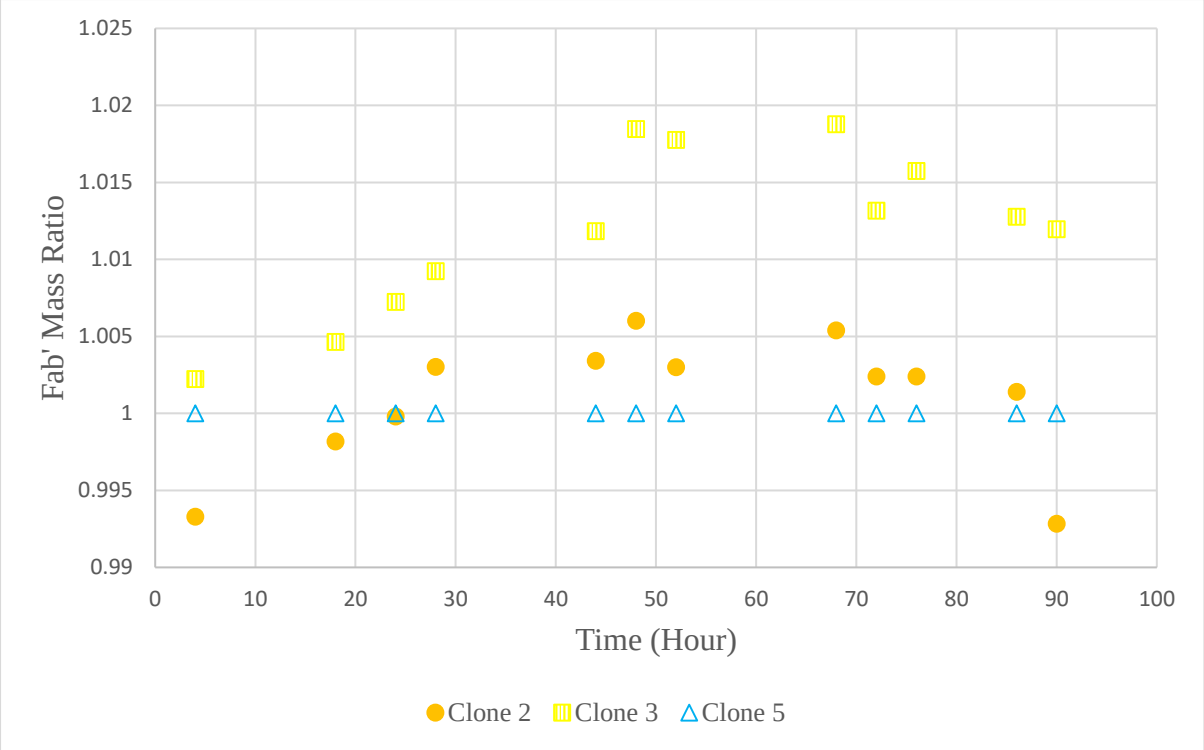


Figure 6.10: Ratio of Fab' Mass for all three clones normalised to the lower of the three, Clone 3, in this instance

Production at large-scale priorities the titre achieved, therefore, the evaluation of the clones should be based solely on the total Fab' weight yield. When comparing Figure 6.4.3A to Figure 6.4.2B in the previous section, it can be noted that the three clones performs the same in both large and small scale relative to each other; With Clone 3 reaching the highest total Fab' weight, and Clone 2 and Clone 5 achieving similar yields.

This suggests that the optimisation work performed using the TECAN can be extrapolated and applied to larger cultivation vessels. The different clones' relative performances remain the same when scaled-up.

6.5 Chapter Conclusion

The Anti-hepatitis B Fab' sequence was optimised for expression in yeast cells. The genetic sequence expressing the Fab' was written and cloned into pJ902-15 backbone. This plasmid was linearised and successfully transformed into GS115 *P. pastoris* cells creating the QT-GAHT strain, inducible by methanol. This strain is capable of producing the intended Fab' at 800µL, 50mL and 200mL scale. It was shown using three clones that the relative ranking of the clones remains the same from 800µL to 200mL.

Chapter 7: Optimisation of Scale up

The scale up results collected previously in Chapter 5 and Chapter 6 for *E. coli* and *P. pastoris* are summarised in this chapter. The graphs generated from the scale up from 96-well plates to 200mL DASGIP bioreactors are compared side to side. By evaluating the specific yield and total yield at bothscales, it will help ascertain which metric at the 800µL scale would generate a more accurate clonal ranking prediction for the 200mL scale. This chapter will also provide a collective insight into the percentage error associated with the different fermentation scale.

7.1 Use of Specific Yield to Better Predict Clones' Behaviour in a Bioreactor

In bioprocess development, the goal is the optimise for the highest achievable titres. This equates to the total Fab' mass produced in the DASGIP bioreactor. Although final titre is used to evaluate the efficiency of the bioreactor, the same evaluation when imposed on the microscale does not accurately represent clone productivity.

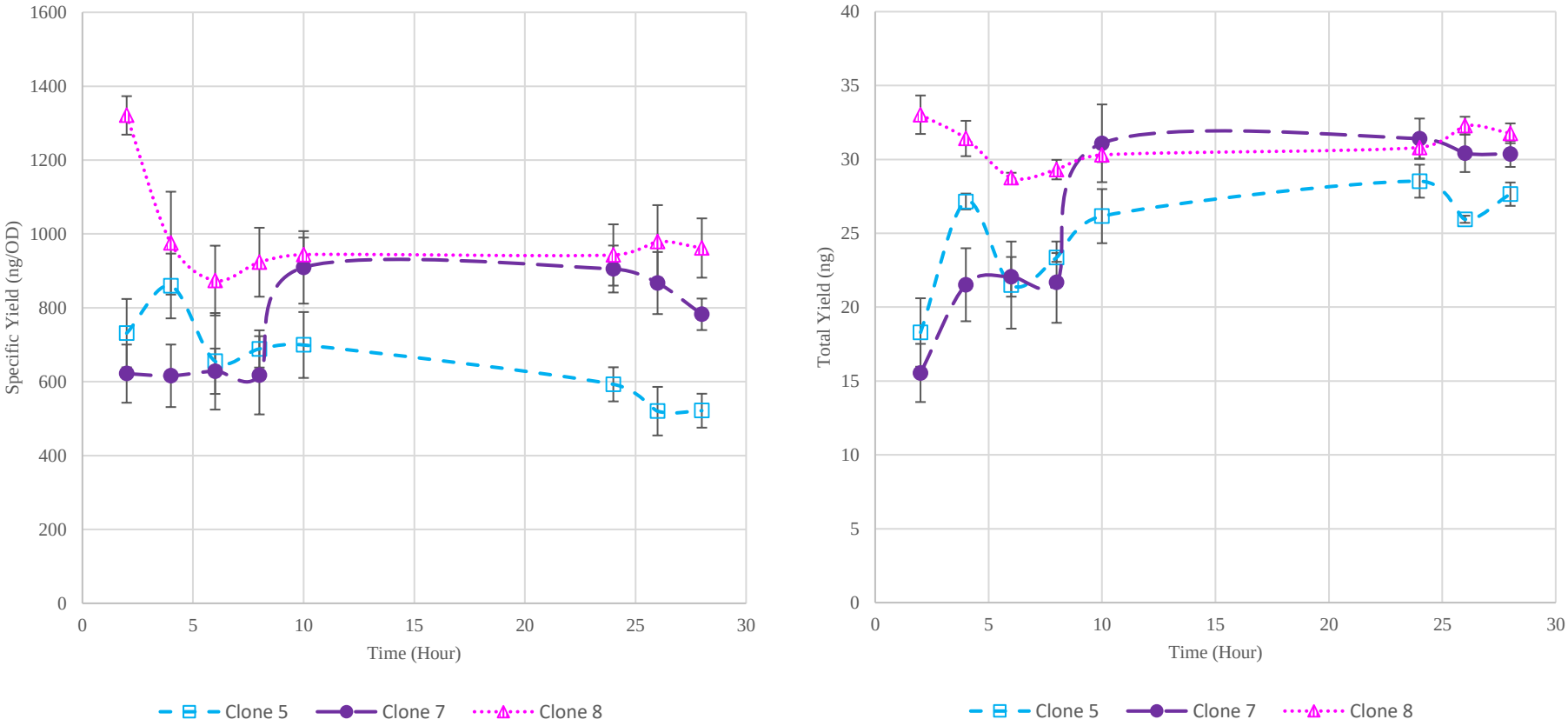


Figure 7.1: The specific yield (Left) and total yield (Right) for *E. coli* clones cultivated in 800µL wells.

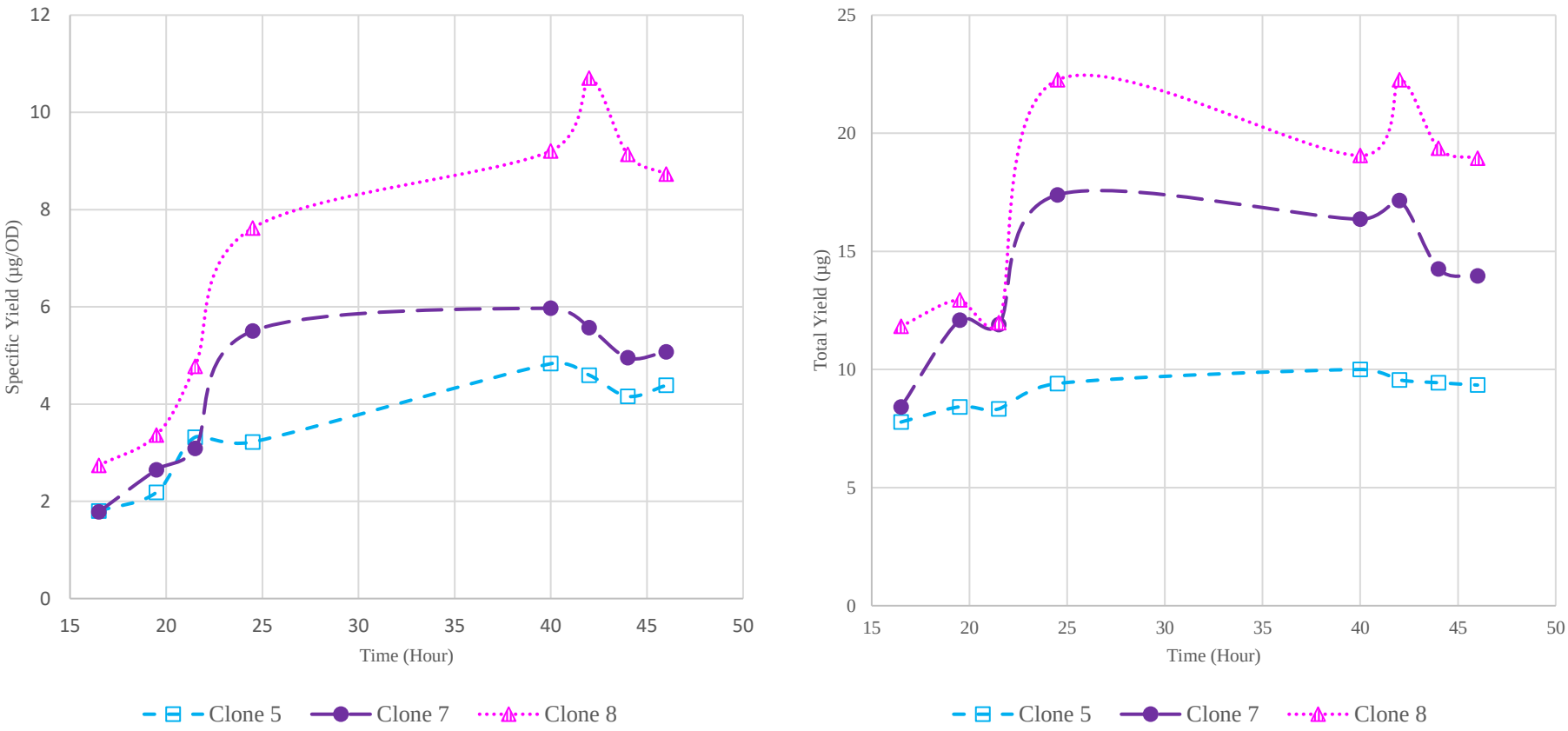


Figure 7.2: The specific yield (Left) and total yield (Right) for *E. coli* clones cultivated in 200mL bioreactors.

In the 800µL microscale cultivation the total yield achieved by Clone 7 is higher than Clone 8. However, in the 200mL bioreactor cultivation, the total yield achieved by Clone 8 is significantly higher than Clone 7. Thus, this is not an accurate representation of the behaviour of clones for commercial scale. Observing the graph for specific yield, which is calculated by dividing the total yield by the OD, the productivity profile much closely resembles the productivity profile at the 200mL bioreactor scale.

This page is intentionally left blank.

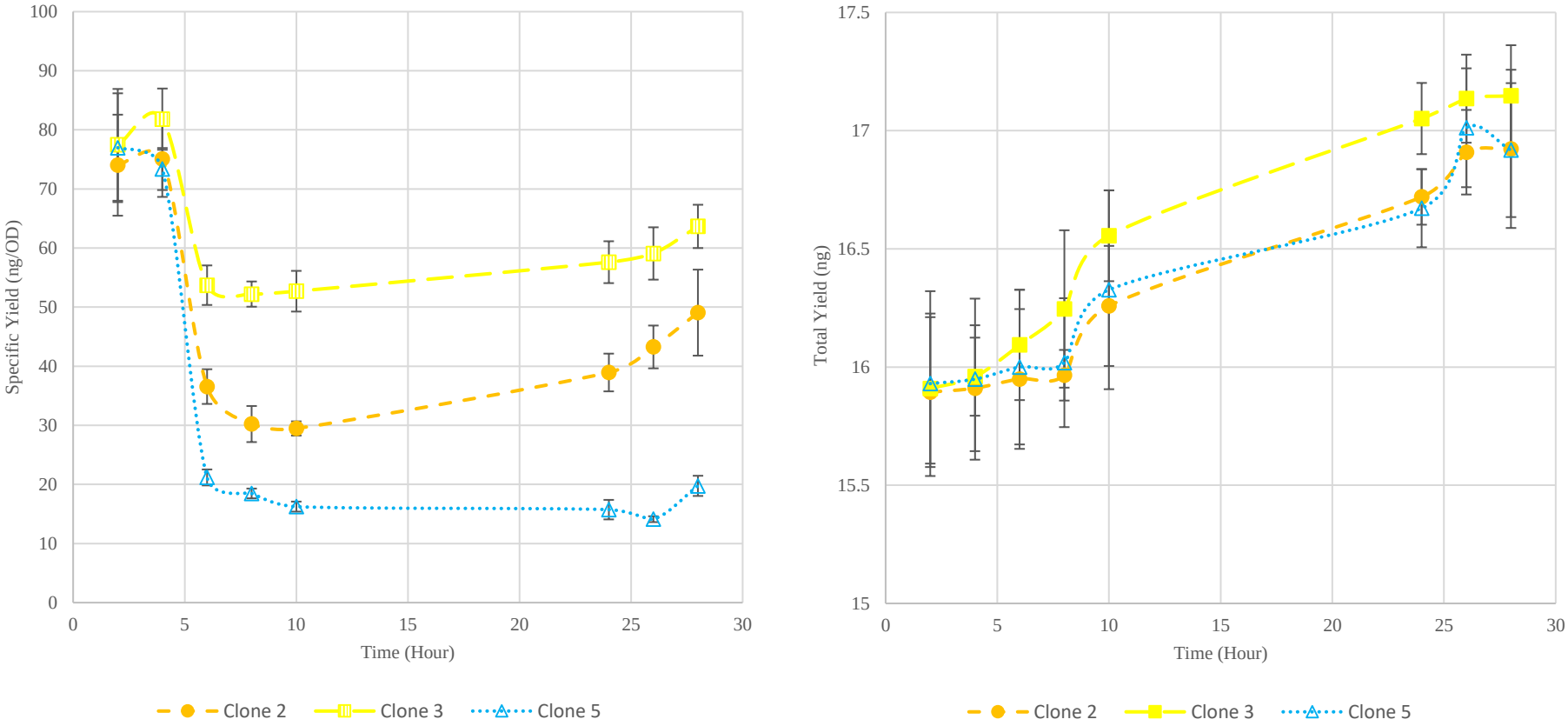


Figure 7.3: The specific yield (Left) and total yield (Right) for *P. pastoris* clones cultivated in 800µL wells.

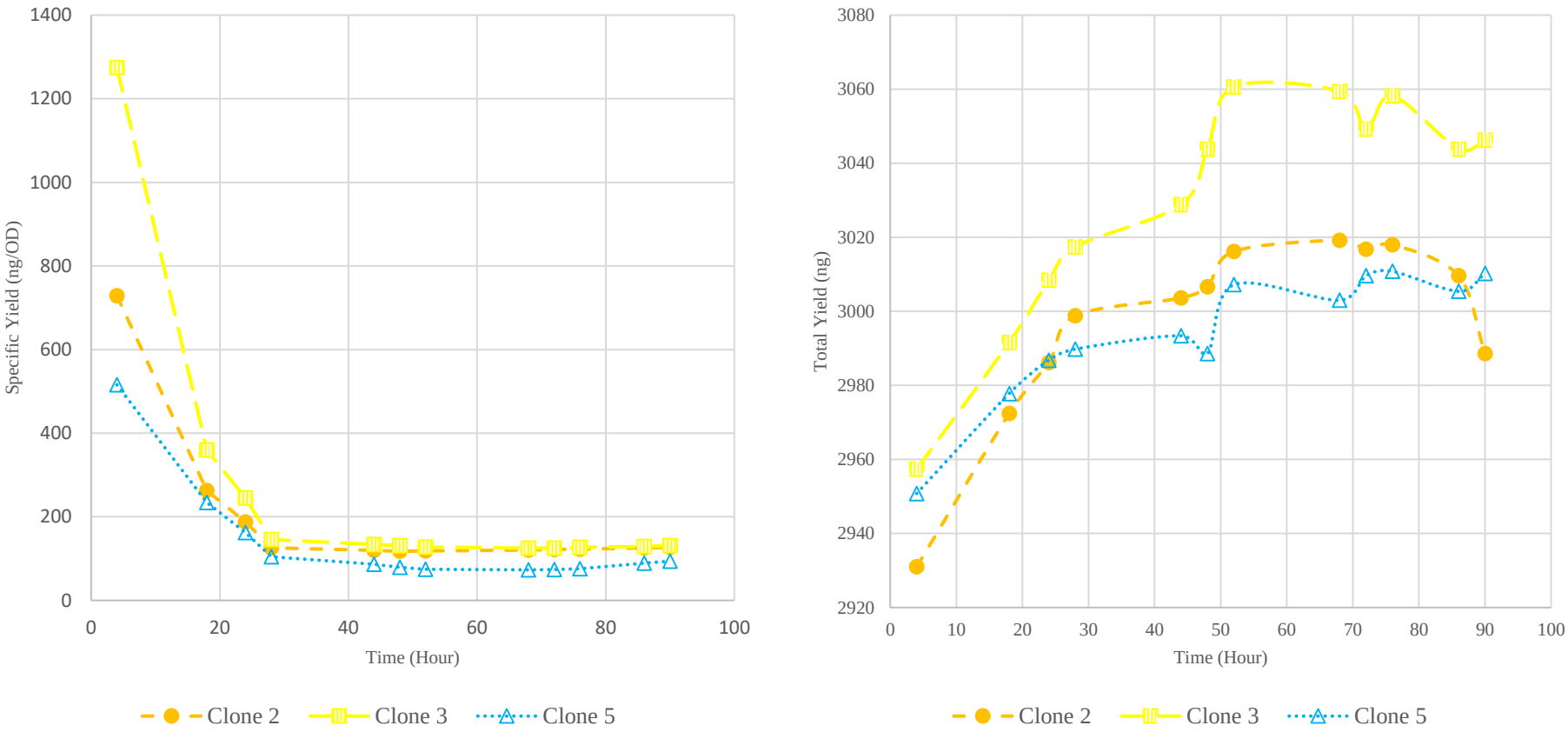


Figure 7.4: The specific yield (Left) and total yield (Right) for *P. pastoris* clones cultivated in 200mL bioreactors.

The scale-up results do for *P. pastoris* is more straightforward. Clone 3 clearly achieves the highest total yield in all four graphs. However, the total yield produced by Clone 2 and Clone 5 in 800µL scale appears to achieve the same amount. When scaled up to 200mL bioreactors, Clone 2 performs better than Clone 5. The stationary phase of the productivity profile at 200mL is better represented by relative Fab' mass at 800µL than by the total Fab' mass at 800µL.

7.2 Standard Deviations of Measurements at 800µL Wells and 200mL Bioreactors

For both *E. coli* and *P. pastoris* the error bars at 800µL scale is much more significant than at 200mL. The errors at 200mL DASGIP bioreactors cannot be perceived. The average percentage error for all the data points for *P. pastoris* at 800µL is 1.5% whereas the average percentage error at 200mL is 0.4%. This is most likely resultant of the lower working volume. Any mild disparity between each well would have a greater effect on the productivity, resulting in larger error bars.

7.3 Chapter Conclusion

Specific yield at the small scale of 800µL is a better predictor of clone performance at the larger scale of 200mL. At the small scale, oxygen saturation, and thereby cell growth reflected as OD, is the limiting factor in Fab' production. With diffusion from a small surface area in the well as the sole source of oxygen, at this scale the OD struggles to reach the magnitude of what is achievable of at an industrial scale. When oxygen limitation removed at a larger scale, clone performance might vary. By ranking the clones using Fab' produced per cell, it gives a more objective view as to how each individual clone performs overall. This is variable that would remain constant regardless of the scale. While, error bars are larger at the smaller scale, they are still within a reasonable percentage range.

Chapter 8: MIAMI for Clone Selection in Bioprocess Development

8.1 Introduction to the MIAMI Concept

Individual cells can express different genes and resulting in phenotypic variations (Čepl, et al., 2016). As a result, different clones of the same strain act differently during cultivation. Although these phenotypes could be hereditary and can be identified, it is potentially a long and extensive process. This process is intensive and costly for optimisation intended for rapid manufacturing.

The creation of MIAMI circumvents the need to identify and classify each clone's inherited phenotypes. Rather, it provides a HT method for evaluating a selection of different clones against specified design criteria. MIAMI targets clone ranking as an objective, with choosing the best performing clone as its primary function. Additionally, it allows for clone selection to be scaled up to screen for multitude of clones in parallel, resulting in a streamlined process to identify the best performing clone.

Clone selection in MIAMI is based on two objective functions. The first objective function is identifying undesired behaviour of the clone and removing the unwanted clonal variation. The second objective function is to then rank the clones from the remaining pool of desired clonal variation based upon their cultivation performance. The intelligence of MIAMI would manifest itself by the appropriateness of its clonal ranking.

8.2 Architecture of MIAMI

MIAMI is the collection of multiple intelligent agents that acts cooperatively to achieve its objective function of clone ranking. MIAMI comprises of two types of agents, assay agents and ranking agents. The assay agents process raw data and analyse the resulting data to identify and remove unwanted clonal variations. The remaining data is then processed by the ranking agents. The ranking agents scans processed growth data to identify periods of stable yields and select for the top performing clones with respect to the maximum yield achieved and minimal standard deviation.

Currently, there are two assay agents and five ranking agents developed for MIAMI as shown in Figure 8.2A. The two assay agents are for handling raw HPLC and flow cytometry data. The ranking agents consists of three stages for *E. coli* and two stages for *P. pastoris*. Each of these stages is represented by its own intelligent agent because the ranking methods differ. The ranking agent of Stage III is a concept explored in this chapter. While it is included in the architecture, it is not fully developed.

Based upon the user input of data type, HPLC or flow cytometry, strain, *E. coli* or *P. pastoris*, and stage, I, II, or III, MIAMI acts as a coordinator, directing the raw data input to the appropriate assay agent and ranking agent. MIAMI will retrieve relevant defined functions from the defined functions database and provide them to the assay and ranking agents. If the data input is insufficient, MIAMI will communicate with the user, prompting for additional data.

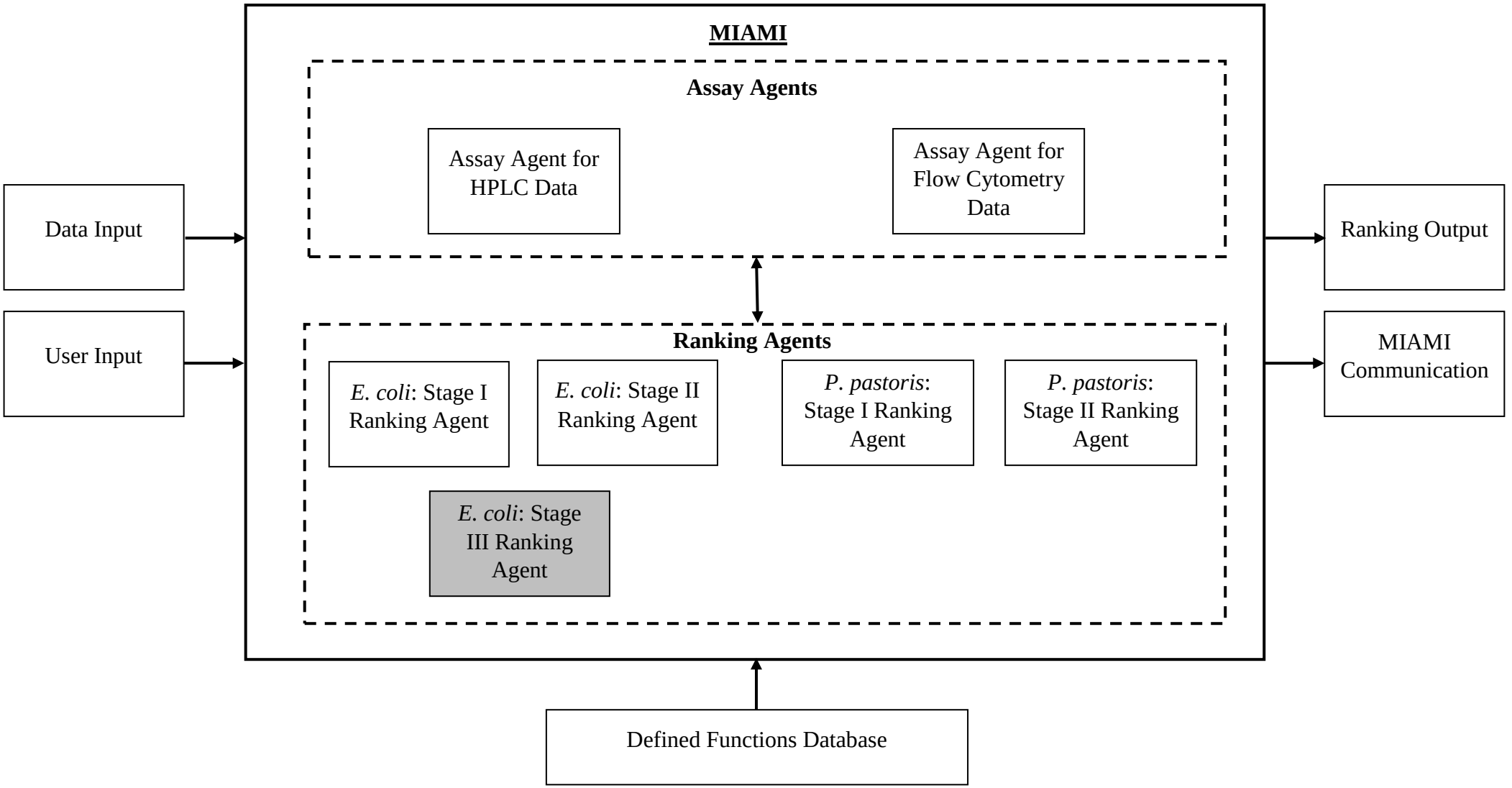


Figure 8.1: Architecture of MIAMI software. Data input consist of the growth and productivity profile of the clones. The user input identifies the data type, strain and stage of the data. MIAMI would identify the appropriate combination of assay agent and ranking agent and direct the data accordingly. The agent would use relevant defined functions from the database to process the raw data and give a ranking output identifying the optimal clone or clones.

8.3 Key Features of MIAMI

8.3.1 User Input of Data Using Excel Spreadsheets

Since OD data can be exported from the plate reader in excel spreadsheet format, all external data is imported into MIAMI using excel spreadsheets. The following code excerpt illustrates how data is imported into MIAMI. In Lines [1] MIAMI will guide to user to an existing excel file for the data to be entered. Line [2] reads the data entered into the excel file and extract the data entered. In this example, the growth profile (OD) of each *E. coli* clones are extracted from the excel file and stored as individual lists in Lines [4] to [11].

```
[1]  print('Please enter values into DATA INPUT ECOLI STAGEI.xlsx.')

[2]  df = pandas.read_excel('DATA INPUT ECOLI STAGE
     I.xlsx',sheet_name=0,index_col=None)
[3]  ODX = df['Time'].tolist()
[4]  OD1A = df['Clone 1'].tolist()
[5]  OD2A = df['Clone 2'].tolist()
[6]  OD3A = df['Clone 3'].tolist()
[7]  OD4A = df['Clone 4'].tolist()
[8]  OD5A = df['Clone 5'].tolist()
[9]  OD6A = df['Clone 6'].tolist()
[10] OD7A = df['Clone 7'].tolist()
[11] OD8A = df['Clone 8'].tolist()
```

8.3.2 Handing of OD Data

Samples prepared for OD reading using the plate reader has a unique challenge of having to deal with dilutions. The linear absorbance range of most plate readers is between 0.1 and 1, thus OD readings larger than 1 would require further dilutions. The samples for each stage are measured at 1X, 4X and 10X dilution factor for *E. coli* and 1X, 10X and 100X dilution factor for *P. pastoris*. The OD reading for all three dilutions would be entered in MIAMI. An excerpt of the code illustrates how MIAMI handles OD data for *E. coli* clones. The full code is available in the database of defined functions in Appendix VII

In Lines [1] to [12] the intelligent agent scans the 10X dilution OD readings and identify if any values are higher than 1. If so, MIAMI would inform the user to perform further dilutions and exit the software in Lines [13] to [14].

```
[1]  if (z_1[0] >= 1 or
[2]  z_1[1] >= 1 or
[3]  z_1[2] >= 1 or
[4]  z_1[3] >= 1 or
[5]  z_1[4] >= 1 or
[6]  z_1[5] >= 1 or
[7]  z_1[6] >= 1 or
[8]  z_1[7] >= 1 or
[9]  z_1[8] >= 1 or
[10] z_1[9] >= 1 or
[11] z_1[10] >= 1 or
[12] z_1[11] >= 1):
[13] print(z_1, "For OD values larger than 1, please perform further dilution")
[14] return exit
```

If no 10X dilution values are larger than 1, the agent proceeds to scan the values of 4X dilutions. If any value is larger than 1, a condition set in Line [1], the intelligent agent will remove the value in Line [2] and insert the factored OD reading at 10X dilution for the sample data point in Line [3]. This process repeats itself for all data points.

```
[1]  if y_1[0] > 1:
[2]  del y[0]
[3]  y.insert(0,z[0])
```

Subsequently, the intelligent agent scans the values of 1X dilutions and replace any values larger than 1 with the OD reading of a higher dilution. As shown in Lines [1] to [3] below. This process also repeats itself for all data points.

```
[1]  if x[0] > 1:
[2]  del x[0]
[3]  x.insert(0,y[0])
```

At the end of the process, all OD values with its appropriate dilutions will be complied within the singular list ‘x’.

8.3.3 Communication from MIAMI

Communication with MIAMI is designed so that the user can double check their data input and critical decisions made by the intelligent agent. Many of the communications from MIAMI asks if the user wants to see the OD, total Fab' yield or relative Fab' yield graphs. An example is depicted below. This generates graphs in the console for the user to inspect.

```
[1]  GraphFB = input("Would you like to see the Fab Graph?")
[2]  if GraphFB == 'yes' or GraphFB == 'Yes':
[3]  print("Great")
[4]  x = np.arange(10)
[5]  pp = plt.plot(FBX,FAB1,color="red",label=CloneA)
[6]  pp = plt.plot(FBX,FAB2,color="green",label=CloneB)
[7]  pp = plt.plot(FBX,FAB3,color="blue",label=CloneC)
[8]  plt.title('Ecoli Stage II Total Fab Produced per Well')
[9]  plt.ylabel('Fab Weight (ug)')
[10] plt.xlabel('Time (Hour)')
[11] plt.legend()
[12] plt.savefig("Ecoli Stage II Fab Graph.png")
[13] plt.show()
```

By showing the graph the user can cross reference the visual depiction of the data sets with MIAMI's ranking.

8.4 Applying MIAMI to Flow Cytometer Data and Mathematical Modelling

Recalling from Chapter 3, the GSAV4 *P. pastoris* strain produced GFP when grown using sorbitol or glycerol as a carbon source. The results, as outlined in the previous chapters, are used to develop the software to handle GFP data. Despite not having a GFP standard to definitively quantify the results, the data can be used to show the trend. If a GFP standard is provided, it can be easily integrated into the existing software.

Working with 96 well plates, the Attune NxT flow cytometer can process plates with the same configuration used in the TECAN. As a result, a streamlined small-scale process was established. The TECAN can be used for cultivation and sample preparations. Then the samples can be transferred to a flow-cytometer for analysis. The data generated from the flow cytometer can processed and plotted to show the productivity profile of the fermentation.

8.4.1 Mathematical Decay Modelling Function for Decay Data Sets

The gene network in the GSAV4 strain creates a biological switch that inhibits protein synthesis when methanol is introduced as a carbon source. The effectiveness of this biological switch was evaluated by monitoring how quickly GFP presences decay. Therefore, a decay modelling function was developed.

The least squares fitting method is used to fit an equation to the decay of GFP data for the GSAV4 strain. The decay function is denoted as Equation (2)

$$y = A \times e^{B \times t} \qquad (2)$$

Equations (3) and (4) are used to determine the exponential constants of the decay function in Equation (2). The coordinates of each data point are represented as x_i and y_i.

$$a = \frac{\sum_{i=1}^{n}(x_i{}^2 y_i)\sum_{i=1}^{n}(y_i \ln y_i) - \sum_{i=1}^{n}(x_i y_i)\sum_{i=1}^{n}(x_i y_i \ln y_i)}{\sum_{i=1}^{n} y_i \sum_{i=1}^{n}(x_i{}^2 y_i) - \left(\sum_{i=1}^{n} x_i y_i\right)^2} \qquad (3)$$

$$b = \frac{\sum_{i=1}^{n} y_i \sum_{i=1}^{n}(x_i y_i \ln y_i) - \sum_{i=1}^{n}(x_i y_i)\sum_{i=1}^{n}(y_i \ln y_i)}{\sum_{i=1}^{n} y_i \sum_{i=1}^{n}(x_i{}^2 y_i) - \left(\sum_{i=1}^{n} x_i y_i\right)^2} \qquad (4)$$

Where

$$A = e^a \qquad (5)$$

$$B = b \qquad (6)$$

These equations are coded into the MIAMI software, and is used to fit decay equation to the GFP results. An excerpt of the code for calculating the constant A is shown below. The code for constant B is written in a similar manner. The code in its entirety is shown in Appendix III.

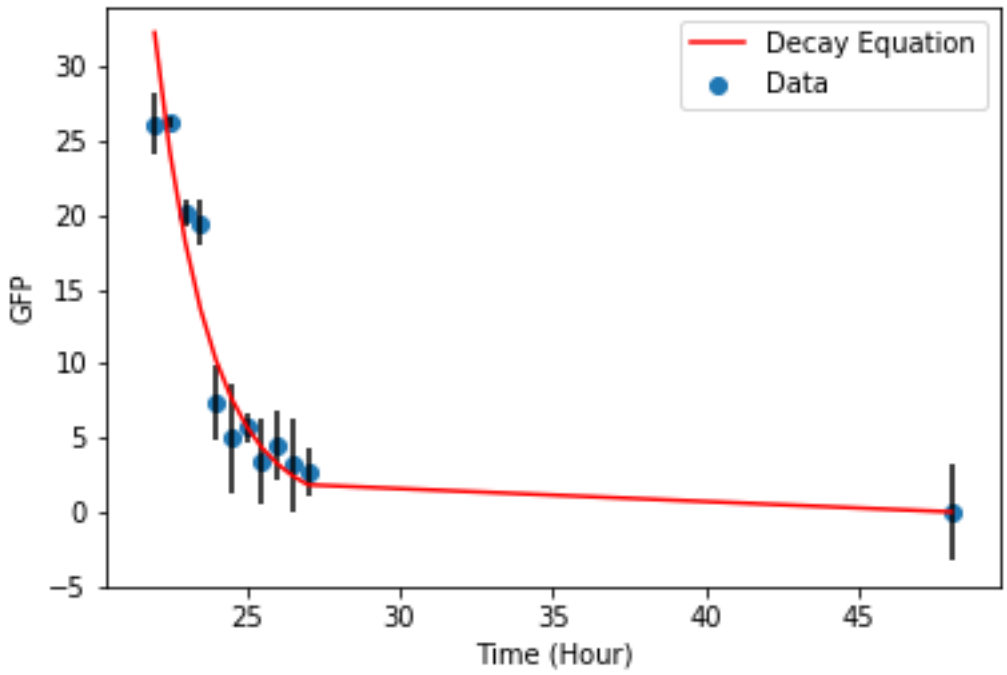
A loop is created to calculate each individual summation operations as denoted in Equation (3) in [1] to [7]. After reaching the upper limits of the operation, the code exits the loop and calculates the values ‘a’ and ‘A’ in [8] and [9] respectively.

```
[1]  while n >= i_1:
[2]  sum_xxy = sum_xxy + x[int(i_1)] * x[int(i_1)] * y[int(i_1)]
[3]  sum_ylny = sum_ylny + y[int(i_1)] * np.log(y[int(i_1)])
[4]  sum_xy = sum_xy + x[int(i_1)]* y[int(i_1)]
[5]  sum_xylny = sum_xylny + x[int(i_1)]* y[int(i_1)] * np.log(y[int(i_1)])
[6]  sum_y = sum_y + y[int(i_1)]
[7]  i_1 = i_1 + 1
[8]  a = (sum_xxy*sum_ylny - sum_xy*sum_xylny) / (sum_y * sum_xxy - sum_xy *
     sum_xy)
[9]  true_A = np.exp(a)
```

8.4.2 Applying MIAMI's Decay Modelling to a GSAV4 Cultivation Run

GSAV4 was grown in a 96 deep well plate initially in glycerol. After an uninterrupted 20 hours incubation period at 30ºC, samples are taken in duplicates of four every half an hour. At the 22-hour mark, the media is changed from glycerol to methanol. For the subsequent 12 hours, samples are taken in quadruplicates every half an hour. The samples were pelleted and resuspended in PBS using the TECAN, and then transferred to the flow cytometer for analysis. The resulting data shows the disappearance of GFP, resembling what was observed in shake flasks in Chapter 3.

MIAMI was able to use its decay modelling to generate a decay equation to the data set as displayed in Figure 8.4.2A. The decay model generated by MIAMI is befitting of the decay data, with majority of the equation falling within the range of the error bars. This result is not an evaluation on the mathematical methods used for decay modelling, but rather a validation of MIAMI's capability of handing flow cytometer data and its potential to integrate mathematical modelling. This modelling function is stored within the defined function database. In future development, multiple mathematical modelling functions could be utilised to predict clonal behaviour.




```
y= 9877904.2312524 e^( -0.5741098044002614 t)
```

Figure 8.2: Console display of the output of MIAMI, a plot of the decaying flow cytometer data and its decay equation generated. The blue line indicates the GFP decay of GSAV4 when its carbon source is switched to methanol. The decay equation displayed at the bottom is calculated in MIAMI using least square method, and plotted in the same graph in red.

8.5 Developing MIAMI for HPLC Data Sets for Clone Ranking

8.5.1 Overview of MIAMI's Operations

There are three primary operations for cultivation clone optimisation designed in MIAMI as shown in Figure 8.5.1A. Stage I, where three top performing clones are selected from a pool of eight. Stage II, where the three previously selected clones are grown in quadruplicates. Sampling measurements are taken to select for one optimal clone. Lastly, in Stage III, which has only been developed for *E. coli*, the top performing strain will be induced at different points in time and an optimal induction time would be calculated.

Table 8.1: Summary of the intended operations in stage of MIAMI for the three different strains. The stages highlighted in green are accomplished, and those highlighted in red are not completed.

Strain	Stage	Operation Description
E. coli	I	Grow 8 different clones for 34 hours and choosing the top 3 performing clones based on titre.
	II	Grow the top 3 performing clones for 28 hours and selecting the top performing clone based on titre and stability.
	III	Grow the top performing clone and inducing with IPTG at different time points for 20 hours. Identifying the optimal induction time-point and induction OD.
P. pastoris	I	Grow 8 different clones for 96 hours and choosing the top 3 performing clones based on titre.
	II	Grow the top 3 performing clones for 28 hours and selecting the top performing clone based on titre and stability.
	III	Grow the top performing clone and inducing with methanol at different time points. Identifying the optimal induction time-point and induction OD.
CHO	I	Grow 8 different clones and choosing the top 3 performing clones based on titre.
	II	Grow the top 3 performing clones and selecting the top performing clone based on titre and stability.
	X	X

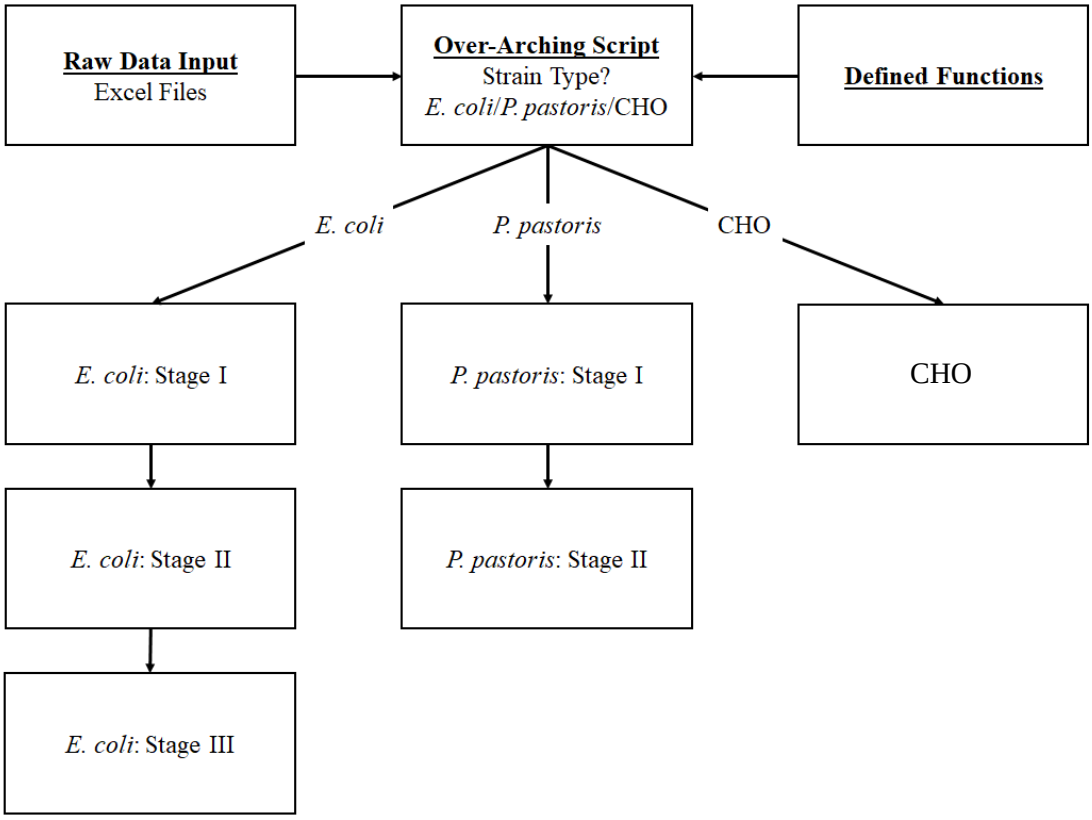


Figure 8.3: Basic flow chart of the interface of MIAMI for strain selection. The over-arching script passes the raw data and relevant defined functions to the correct branch, either *E. coli* or *P. pastoris*, and the clones are evaluated throughout the stages.

MIAMI code quantifies the considerations made as a human user to simulate the same decision-making process. The intricacies of the code in each stage draws information from the data and reflect upon the performance of each clone to make an informed decision pertaining to the best yield from fermentation. To support data analysis within the software, various defined functions are designed and stored in a database. The overarching script acts as a coordinator by providing the appropriate functions to the data set to aid analysis.

Moving forward, to correctly quantify different aspects of evaluating cultivation growth and productivity, data previously collected using the TECAN is used to develop the MIAMI software. The goal is to design MIAMI to evaluate the clones to reflect the assessment made in the previous chapter. The simulated data would be used as a standard to fine tune the particulars of the ranking process within each stage. Once the MIAMI software is completed, a run would be conducted to verify the accuracy of the intelligent agents.

8.5.2 *E. coli* Stage I Intelligent Agent

The first stage of clone selection involves a preliminary assessment of a larger sample size of different cell clones. The initial plan was to utilize a deep 96 well plate, with one clone per well. However, there were quirks and limitations of the liquid handling systems that needs to be considered. A significant point to consider is a lack of unique equipment required to run 96 different clones, due to only having one thermomixer unit. Another consideration is the contamination risk of repeated sampling and media top up. This would also increase the likelihood of cross contamination, thus vastly reducing the accuracy of the results collected.

Reflecting on all discussed issues, Stage I of the MIAMI process for *E. coli* would involve a rudimentary map of the growth profile of 8 different clones over a period of 48 hours. The monitoring time gives a better view of the cells' growth, taking into deliberation the time required for a low copy number plasmid generate Fab'.

The MIAMI software is fed the simulated data of both the optical density (OD) and Fab' weight produced. In Stage I, the data is processed to express the total amount of Fab' produced per 800µL well. The resulting productivity profile for each clone is stored as a list within the code. The profile is sorted to identify three of the highest consecutive data points. The average of the highest three data points would then be used to rank the clones.

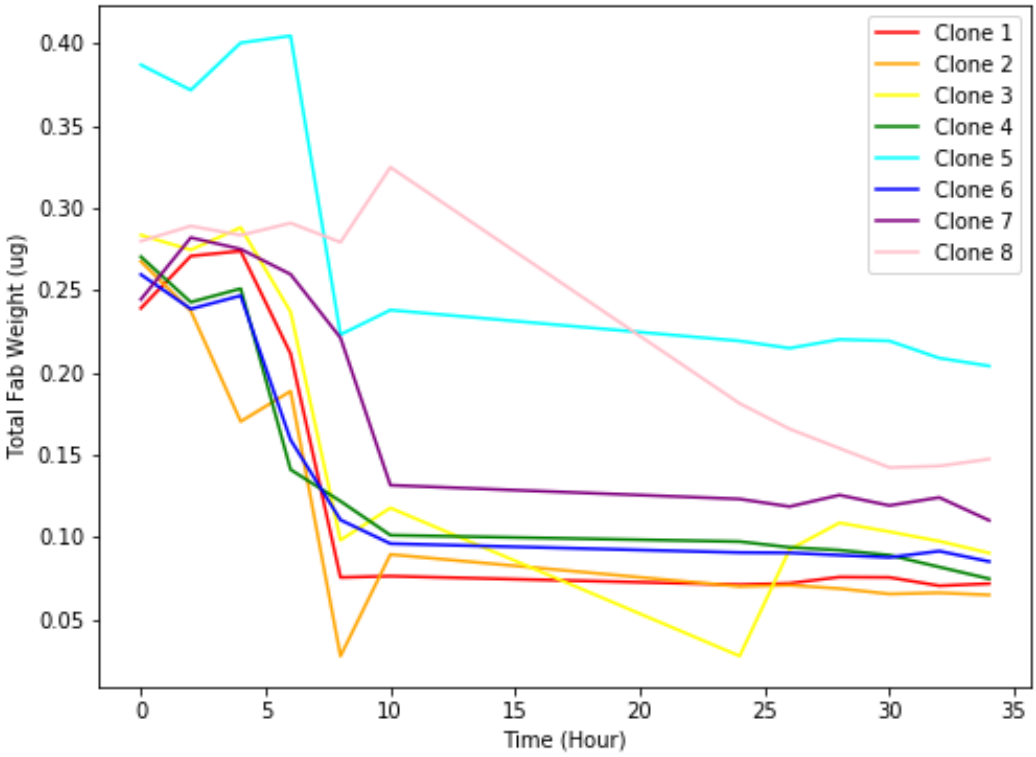


Figure 8.4: Output image generated by MIAMI that depicts the total Fab' produced per 800 µL well for *E. coli* clones.

In Figure 8.5.2A, the clones tend to maintain a stationary period of Fab' amount for approximately 6-8 hours, 3-4 data points, before a significant drop. Thus, finding the mean of the highest three consecutive data points would yield the average amount of Fab' during the stationary period. The use of the mean of three different points would also safeguard against selecting an outlier in error. An excerpt of the code is shown to illustrate how the performance a clone is ranked at Stage I. The code in its entirety is shown in Appendix IV.

8.5.2.1 *E. coli* Stage I Sorting Algorithm

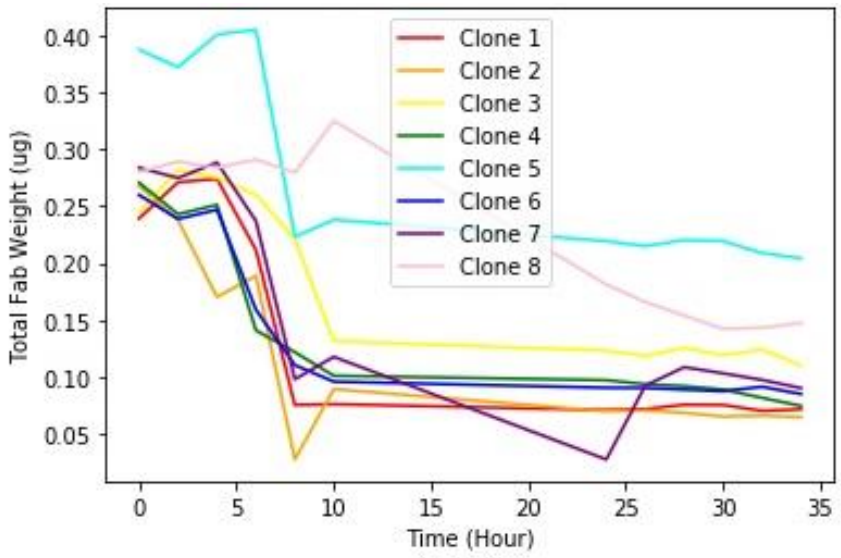
Line [1] uses a sorting algorithm to scan the growth profile and calculate the average yield in a period of three data points, such that it returns the average of data points 1, 2, and 3, data points 2, 3, and 4, data points 3, 4, and 5, etc. The average consecutive yields of Clone 1 are calculated and stored as a list named FBM1. Line [2] sorts the list FBM1 from lowest variable to highest. Line [3] selects the highest consecutive yields of Clone 1 and stores that value as FABM1. This code is repeated for all clones to find their highest consecutive yield.

```
[1]  FBM1 = dfc.List_Conseq_Sort_3X12(FAB1)
[2]  FBM1.sort()
[3]  FABM1 = FBM1[-1]
```

8.5.2.2 *E. coli* Stage I Ranking Algorithm

After identifying the highest consecutive yields for all clones, a ranking algorithm enters their yield values and their clone number are entered as an array as shown in Line [1] to [10]. In Line [11] the array is sorted from lowest value to highest, and in Line [12] the three highest values are selected and stored in a separate array. The three top performing clones are displayed in the console output as shown in Figure 8.5.2B.

```
[1]  ecstrainsort = [
[2]  (FABM1, 'Clone 1'),
[3]  (FABM2, 'Clone 2'),
[4]  (FABM3, 'Clone 3'),
[5]  (FABM4, 'Clone 4'),
[6]  (FABM5, 'Clone 5'),
[7]  (FABM6, 'Clone 6'),
[8]  (FABM7, 'Clone 7'),
[9]  (FABM8, 'Clone 8'),
[10] ]
[11] ecstrainsort.sort()
[12] ecTOP3 = (ecstrainsort[-1],ecstrainsort[-2], ecstrainsort[-3])
```




```
Would you like to save the Fab Graph?Yes
The top three strains are:
1st: Clone 5
2nd: Clone 8
3rd: Clone 7
```

Figure 8.5: Console output of the MIAMI software. The ranking of the *E. coli* clones in Stage I, evaluated using MIAMI.

8.5.2.3 Discussion of *E. coli* Stage I Intelligent Agent

The top performing clones in this stage is more straightforward to ascertain because there is only one measurement per data point. A less sophisticated evaluation protocol is adopted at this stage where the ranking looks purely at the total amount of Fab' produced per well. With only one reading per data point, inclusion of more variables would lead to additional sources of error. Hence, by plotting purely the Fab' data, it means the graph generated a more precise reflection of the productivity, with any associated error can be attributed to sample extracted for the HPLC. While the OD is not used directly in this stage in the evaluation process, the raw data is still processed. A graph can be generated for the user to inspect.

As seen in Figure 8.5.2.2B, the intelligent agent selects three of the top performing clones to do more comprehensive growth profiling in Stage II. Selecting three clones rather than one reduces the chance of overlooking a good strain due to any errors.

8.5.3 *P. pastoris* Stage I Intelligent Agent

P. pastoris cells have a longer doubling time, thus the growth of these cells is initially monitored over 96 hours. *P. pastoris* cells secret the Fab' directly into the media, thus, no extraction method is needed to harvest the Fab'. The total amount of Fab' produced should therefore have a positive trajectory. The existing data shows that the Fab' produced for certain clones, such as Clone 1, decreases over time, which is not how the growth profile is expected to behave. Thus, during manual selection, these clones are discounted. This selection criterion is unique for this *P. pastoris* strain and would be crucial to address it as part of the *P. pastoris* intelligent agent. A scanning algorithm is therefore created for the analysis of *P. pastoris* clones.

8.5.3.1 *P. pastoris* Stage I Scanning Algorithm

The scanning algorithm scans the first three quarters of the yield profile and tries to remove the clones that do not display an upwards trend. The last quarter of the yield profile is discounted at this stage because after three days of cultivation and sampling there was a higher chance of cross contamination and error. The more erratic behaviour of the yield can be seen in Figure 8.5.3.1A, with more dramatic increases and decreases after 72 hours.

Theoretically, the amount of Fab' produced should always be increasing, and slowly reaching an asymptotic maximum as the cells die and Fab' stops being produced all together. Thus, the average of the middle section of the Fab' profile should always be higher than the average of the first quarter. Using the average ensures that any outlier due to error would not significantly affect during the evaluation process.

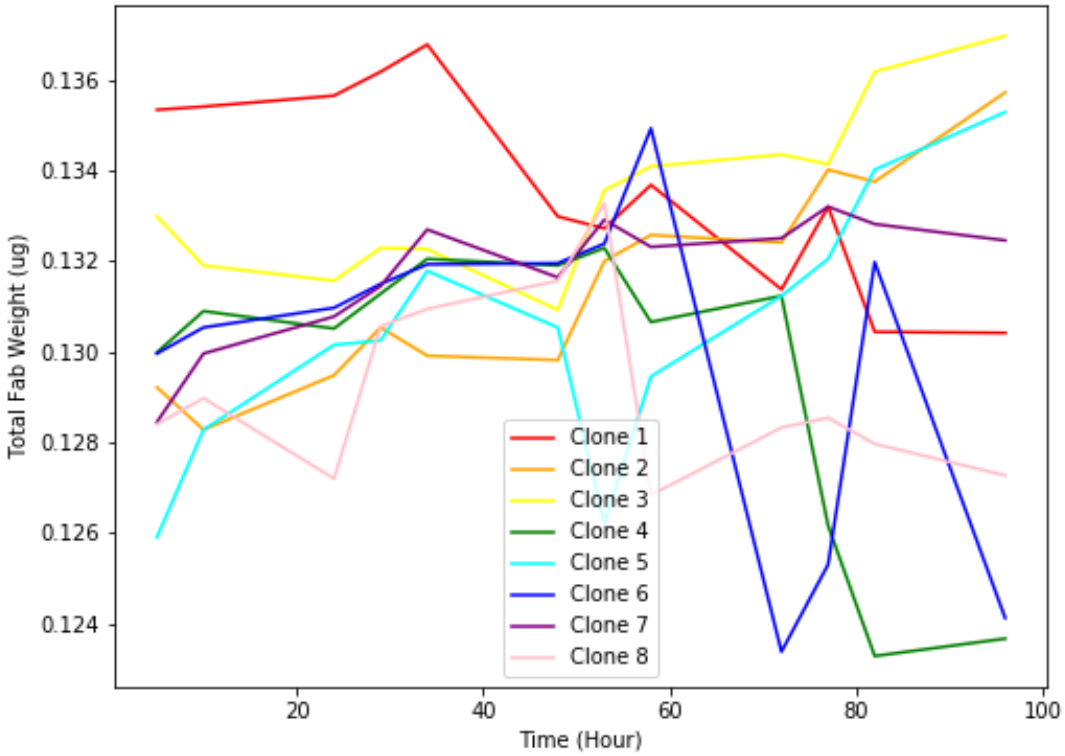


Figure 8.6: Graph produced by MIAMI that depicts the total Fab' produced per 800µL well for *P. pastoris* clones.

An excerpt of the code is shown below to illustrate how the scanning algorithm discounts unwanted behaviour for *P. pastoris* clones. The full code is shown in Appendix V. The full profile of each clone and their clone numbers are initially all stored as one array named 'all_values'.

```
[1]  all_values = [(a,"Clone 1"),
[2]  (b,"Clone 2"),
[3]  (c,"Clone 3"),
[4]  (d,"Clone 4"),
[5]  (e,"Clone 5"),
[6]  (f,"Clone 6"),
[7]  (g,"Clone 7"),
[8]  (h,"Clone 8")]
```

Line [1] calculates the average Fab' yield for Clone 1 from hours 0-24. Line [2] calculates the average Fab' yield for Clone 1 from hours 24-72. This process is repeated for all *P. pastoris* clones.

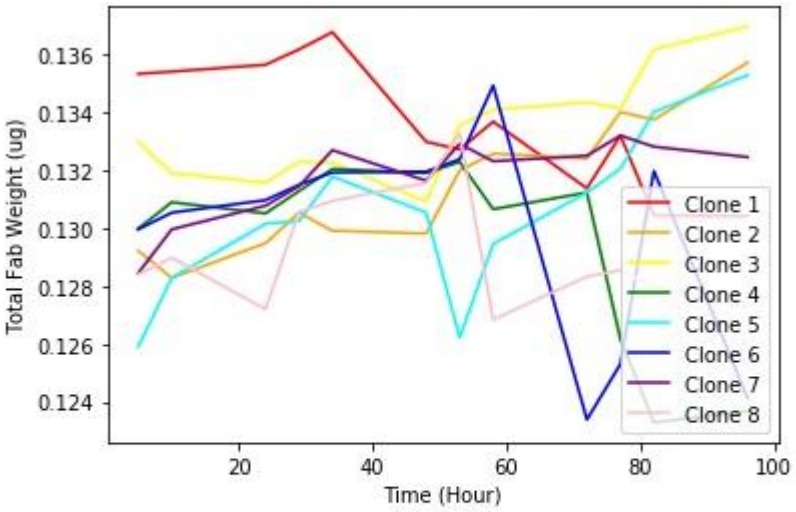
```
[1]  a_3 = np.mean([a[0], a[1], a[2]])
[2]  a_6 = np.mean([a[3], a[4], a[5],a[6], a[7], a[8]])
```

If the average yield at hours 24-72 is lower than the average at 0-24, the intelligent agent will remove the data set from the initial array. If the condition in Line [1] is met, then the growth profiles will be removed from the array 'all_values' in Line [2].

```
[1]  if a_3 > a_6:
[2]  del all_values[0]
```

8.5.3.2 Discussion of P. pastoris Stage I Intelligent Agent

The clones that were not removed by the intelligent agent will then be ranked using the same sorting and ranking algorithm for *E. coli* Stage I as described in Section 8.5.2.1 and 8.5.2.2. As a result of the intelligent agent removing unwanted clonal behaviour, Clone 1 was not considered for the top three performing clones. Although it clearly had the highest yield at the beginning, as shown in Figure 8.5.3.2B, the yield decreased over the course of cultivation. The intelligent agent was able to select clones with an overall positive trend in yield.




```
Would you like to save the Fab Graph?yes
The top three strains are:
1st: Clone 3
2nd: Clone 2
3rd: Clone 5
```

Figure 8.7: Console output of the MIAMI software. The ranking of the *P. pastoris* clones in Stage I, evaluated using MIAMI.

8.5.4 *E. coli* and *P. pastoris* Stage II Intelligent Agent

This stage performs the assessment of the yield of the top three performing clones identified previously in quadruplicates. The Stage II intelligent agent for *E. coli* and *P. pastoris* share the same algorithms and thus would be discussed jointly in this section. Unlike in Stage I, each data point in their OD and productivity profile is the mean of four readings. The standard deviation of each point is used as an indication of variance in cultivation performance. Therefore, ranking of the clones in this stage takes into consideration of both the maximum amount of Fab' achieved, and the variances of the data set. Excerpts of the code is shown below to illustrate the ranking process.

The scale-up discussions in Chapter 7 explained ranking specific yield at 800μL scale for each clone is a more accurate representation of how the clone would perform at a larger scale. Thus, when processing raw data, the assay agent would have to calculate the specific yield of each clone by dividing the total yield by OD for each data point. Line [1] to [3] shows this operation for all three clones screened at Stage II.

```
[1]  FBOD1 = dfc.List_Divide_8(FAB1,OD1)
[2]  FBOD2 = dfc.List_Divide_8(FAB2,OD2)
[3]  FBOD3 = dfc.List_Divide_8(FAB3,OD3)
```

8.5.4.1 *E. coli* and *P. pastoris* Stage II Sorting Algorithm

The sorting algorithm scans the growth profile and calculates the average yield in a period of three data points and their associated variability. These are the two variables that would be used to determine the clone's ranking.

8.5.4.2 *E. coli* and *P. pastoris* Stage II Ranking Algorithm

The ranking algorithm in Stage I is very straightforward, since it only takes into consideration for one variable. However, in Stage II a more sophisticated method is utilised. The raw values for stable yield would generally be at least an order of magnitude larger than its variance. To be able to evaluate the clones based on two vastly different factors, highest stable yield and variance, the data must be standardised in a way where the variables can be directly comparable.

MIAMI determines an average maximum yield based upon the growth profile. Each maximum yield is compared to the remaining data set and is assigned a value indicating its relative performance. This value is represented by the ratio of the individual yield and the average yield of all runs. Using the same method, an inverse value is derived to signify the variance of the yield. The code used to standardise the highest stable yield is shown below. Lines [1] to [3] sorts the average values of the three clones and identifies the highest value and stores it as the variable 'avg_m'. Lines [4] to [6] calculates the highest stable yield of each clone as a percentage of the highest value. Variance of each clone is standardised in the same manner. Using this operation, both factors will be standardised and represented as a percentage of the maximum highest stable yield and minimum variance achieved.

```
[1]  all_avg = [a_avg,b_avg,c_avg]
[2]  all_avg.sort()
[3]  avg_m = all_avg[-1]

[4]  a_avgp = a_avg / avg_m
[5]  b_avgp = b_avg / avg_m
[6]  c_avgp = c_avg / avg_m
```

The default weight on the variables, productivity and variance, by the ranking algorithm in Stage II is 75% and 25% respectively. An excerpt of the code that demonstrates this is shown below. The entirety of the code is shown in Appendix V. Lines [1] to [3] adds the default 75% and 25% weighting to the standardised values of highest stable yield and variance.

```
[1]  Val1 = 0.75*ValM1 + 0.25*Err1
[2]  Val2 = 0.75*ValM2 + 0.25*Err2
[3]  Val3 = 0.75*ValM3 + 0.25*Err3
```

Table 8.5.4.2A and Table 8.5.4.2B presents the ranking values of using average values directly and using relative values as discuss. Directly using the average values, the variance of the data set has little to no impact on the resulting ranking value. However, when using relative values, the low variance of Clone C allowed for it to be ranked higher than Clone B.

Table 8.2: Fictitious example data to illustrate *E. coli* and *P. pastoris* Stage II ranking algorithm results when using average values.

	Average Stable Yield	**Average Variance**	**Inverse Variance**	**Ranking Value**
Clone A	10	1.0	1.0	7.75
Clone B	14	1.5	0.67	10.67
Clone C	13	0.8	1.25	10.06

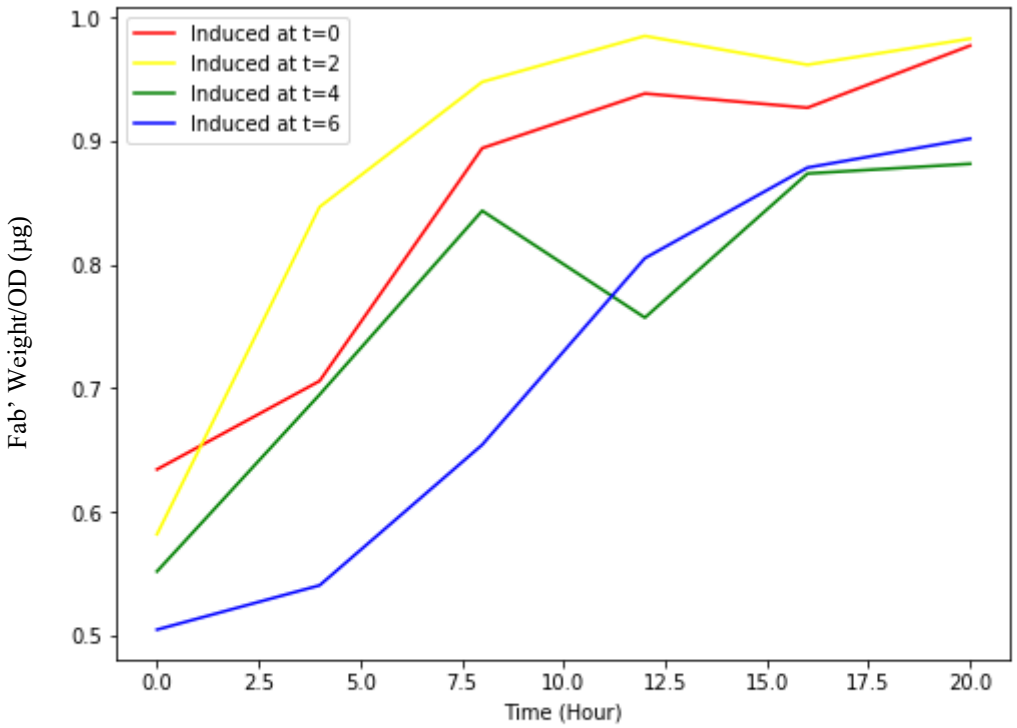
Table 8.3: Fictitious example data to illustrate *E. coli* and *P. pastoris* Stage II ranking algorithm results when using relative values.

	Relative Stable Yield	**Relative Variance**	**Inverse Variance**	**Ranking Value**
Clone A	0.714	0.667	1.500	0.9105
Clone B	1.000	1.000	1.000	1.0000
Clone C	0.928	0.533	1.875	1.1648

The lesser weight put on the variance is because of the scale at which the cultivation is conducted. At the scale of 800µL, a small difference would have a significant effect on the productivity. From scaling up the cultivation to 250mL DASGIP bioreactors, the margin of error, derived from standard deviation, was reduced. Under the assumption that standard deviations would be less at a large-scale fermentation, MIAMI places less significance in low variances. This allows for the software to not discount a potentially high performing strain because of high error bars. The productivity of the clones is ranked considering the amount of Fab' produced per cell and the associated error. This generates one optimal clone with the best productivity.

8.5.5 *E. coli* Stage III Intelligent Agent

Stage III was developed for *E. coli* clone analysis. It focused on one top performing clone that is determined and verified in the previous stages. The clone is induced at different times to determine which is more optimal. Wells are induced after 0, 2, 4 and 6 hours of cultivation and four samples are taken on a four-hourly basis. Raw data is processed by the assay agent into specific yield in the same manner as Stage II. Evaluating the processed data using the same scanning and ranking agent as described in Stage II, MIAMI identify the optimal time for induction and OD at the time of induction.




```
Would you like to proceed with default weighting? (Yes/No)yes
Proceeding with default.
The top performing strain is induced at t=2 at an OD of 0.22415000000000002
```

Figure 8.8: Console output of the MIAMI software at stage III. The optimal time for induction as evaluated by MIAMI.

8.6 Intelligent Flexible Features Incorporated into MIAMI

8.6.1 Flexibility in the Processing of Raw Data

Another type of communication from MIAMI gives the user the option to use non-default settings. An example being using the default standard curve derived in Section 5.3.2. Line [1] asks the user if they would like to proceed with default values. If the user does not correctly express no, the intelligent agent will proceed with default values, as shown in Lines [10] to [14]. However, if appropriate values were entered for Lines [4] and [5], the intelligent agent will use the entered values to calculate total Fab' yield, as shown in Lines [7] to [9]. Allowing the user to use MIAMI for different products.

```
[1]  Default = input('Would you like to proceed with default? (Yes/No)')
[2]  if Default == "no" or Default == "No":
[3]  print("Please enter values for A and B for standard y=Ax+B")
[4]  A = input("Value for A")
[5]  B = input("Value for B")
[6]  print("Thank You")
[7]  FAB1 = [(i * int(A) + int(B)) * (8/5) for i in FB1]
[8]  FAB2 = [(i * int(A) + int(B)) * (8/5) for i in FB2]
[9]  FAB3 = [(i * int(A) + int(B)) * (8/5) for i in FB3]
[10] else:
[11] print ("Proceeding with default.")
[12] FAB1 = [(i * 0.00003 + 0.0144) * (8/5) for i in FB1]
[13] FAB2 = [(i * 0.00003 + 0.0144) * (8/5) for i in FB2]
[14] FAB3 = [(i * 0.00003 + 0.0144) * (8/5) for i in FB3]
```

8.6.2 Flexibility in the Weighting of Variables

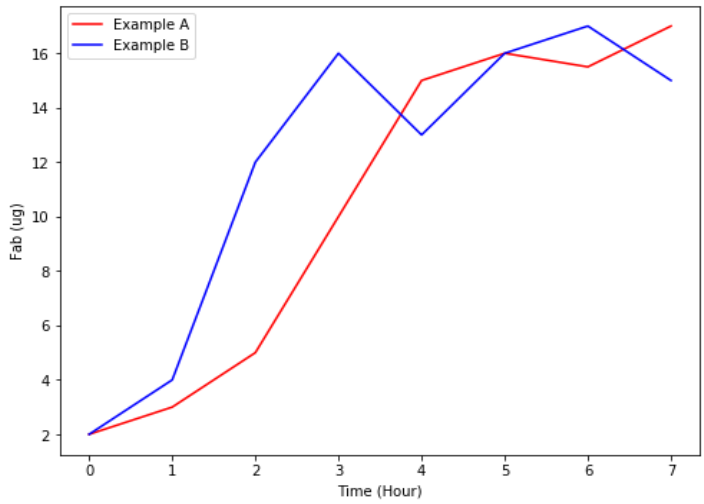
Both Stage II and Stage III includes a feature where different ranking variables, productivity and variance, are weighted by importance. The default setting attributes 75% and 25% to the two variables respectively.

This default weighting can be changed by the user. When a user input for weighting is prompted in Line [1], the user can enter how much weight they want to give the highest stable yield. Line [2] will express this weighting as a percentage for highest stable yield, and Line [3] calculates the weighting for variance. Lines [4] to [6] then applies the user weighting to the standardise values of highest stable yield and variance.

```
[1]  Weighting = input("Productivity Weighting?")
[2]  W1 = int(Weighting)/100
[3]  W2 = 1-W1
[4]  Val1 = W1*ValM1 + W2*Err1
[5]  Val2 = W1*ValM2 + W2*Err2
[6]  Val3 = W1*ValM3 + W2*Err3
```

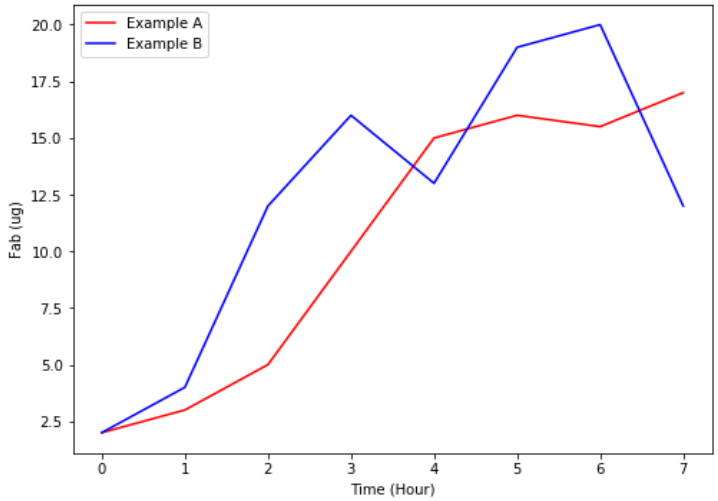
The following graphs are generated using fictitious data to illustrate the variability in results when putting different weighting on the two factors.

Time (Hour)	Example A (µg)	Example B (µg)
0	2.0	2.0
1	3.0	4.0
2	5.0	12.0
3	10.0	16.0
4	15.0	13.0
5	16.0	16.2
6	15.0	17.0
7	17.0	15.5



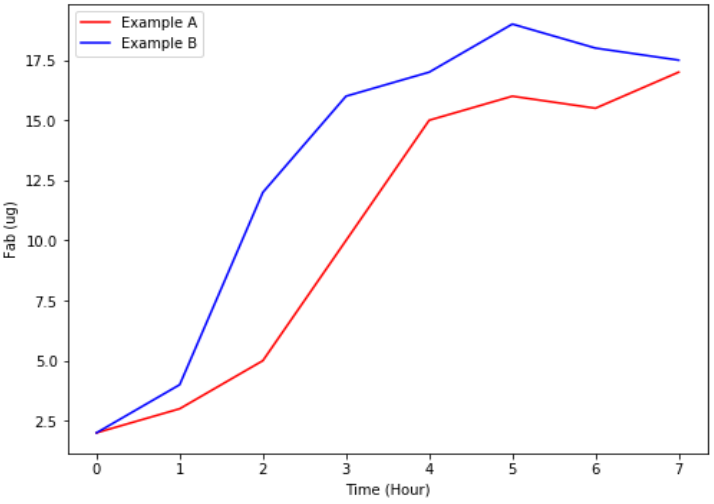

```
The top performing example is Example A
```

Time (Hour)	Example A (µg)	Example B (µg)
0	2.0	2.0
1	3.0	4.0
2	5.0	12.0
3	10.0	16.0
4	15.0	13.0
5	16.0	19.0
6	15.5	20.0
7	17.0	12.0




```
The top performing example is Example B
```

Time (Hour)	Example A (µg)	Example B (µg)
0	2.0	2.0
1	3.0	4.0
2	5.0	12.0
3	10.0	16.0
4	15.0	17.0
5	16.0	19.0
6	15.5	18.0
7	17.0	17.5



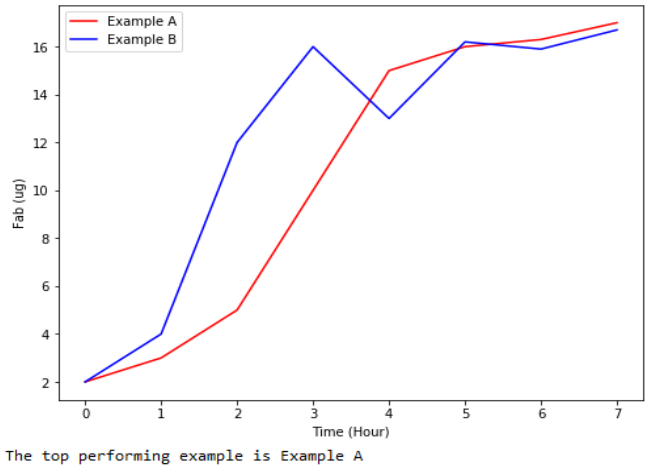

```
The top performing example is Example B
```

Figure 8.9: Three sets of fictitious data to illustrate how the weighted variable feature, at default settings, works in the MIAMI software. MIAMI does not always choose the clone that achieves the highest titer if the variance is high.

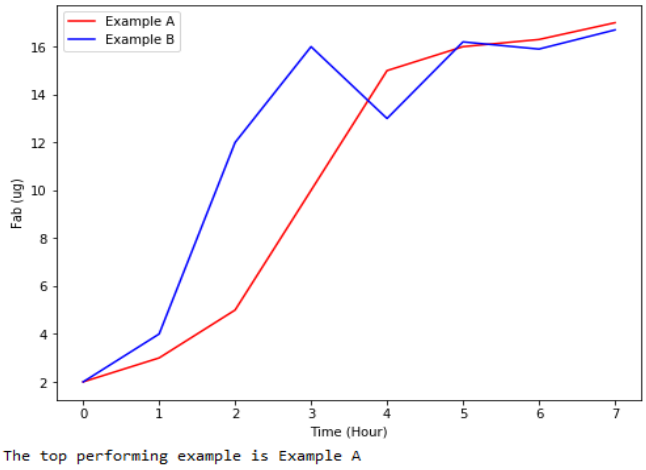
As seen in Figure 8.6.2A (Top), even though Example B reached a higher stable Fab' yield than Example A, the MIAMI software ranked Example A as the top performing example. This is the manifestation of the software considering both the productivity and variance of the dataset. However, the weighting of the variables favours higher productivity rather than lower standard deviation. In the second set of examples in Figure 8.6.2A (Middle), the software prioritises Example B despite the highly variable Fab' data values. In the last set of examples, at similar variances, the software would obviously select the data set with higher productivity.

Prior to ranking, the MIAMI software asks if the user is happy to proceed with the default settings. This feature is added to allow the user to have the flexibility the change the priority of the ranking. As depicted in Figure 8.6.2B, the same sample data is provided to the software, and by changing the weight on productivity and variances yields different results.

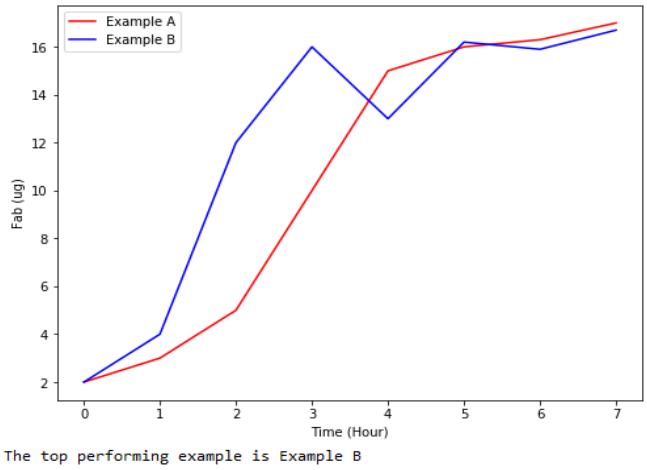
Time (Hour)	Example A (µg)	Example B (µg)
0	2.0	2.0
1	3.0	4.0
2	5.0	12.0
3	10.0	16.0
4	15.0	13.0
5	16.0	16.2
6	16.3	15.9
7	17.0	16.7



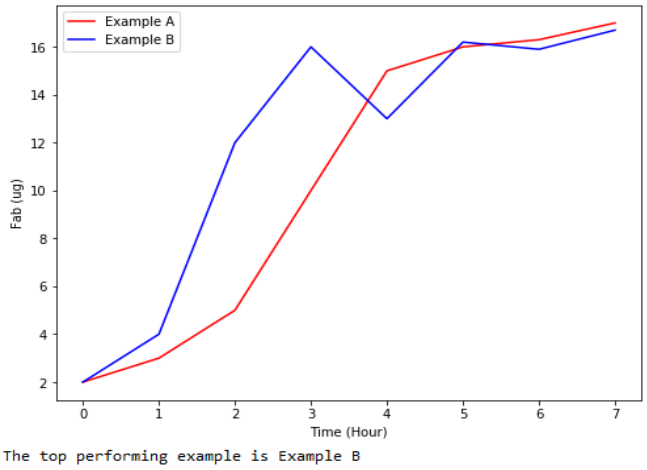
The top performing example is Example A



The top performing example is Example A



The top performing example is Example B



The top performing example is Example B

Figure 8.10: The same set of fictitious data is provided to the MIAMI software. By altering the weight placed on productivity and variances, the software will select a different optimal example. 75% Productivity and 25% Variance (Top Right) and 50% Productivity and 50% Variance (Top Left) favours Example A. Whereas 25% Productivity and 75% Variance (Bottom Right) and 0% Productivity and 100% Variance (Bottom Left) favours Example B.

8.6.3 Intelligent Assessment of Viable Data

To add an intelligent aspect to the MIAMI software, a significant amount of effort has been put into developing layers of complexity when it comes to data analysis. As discussed in the previous sections, the software has inbuilt ways that deals with different types of cells in a suiting manner. For example, treating strains that are susceptible to product loss with a prioritisation on harvesting before cell rupture like the *E. coli* strain that secretes into the periplasm.

8.6.4 Modular Setup

A challenge for the software of IAPBD developed by Wu & Zhou, 2014, as described in section 1.12.2, was the update of the TECAN software from Gemini to EVOware. This was a significant consideration to take onboard going forward. To avoid similar issues when it comes to further software development, a modular approach will be taken and individual agent with specialised functionalities will be developed separately and then subsequently integrated.

MIAMI is designed to handle data processing and evaluation, but not the remote control of automated equipment and automated data retrieval. While a fully integrated and automated link between the MIAMI software and the TECAN, HPLC and flow cytometer is not addressed in this project, it is a possible development. Since software that is capable of this feature had already been developed and used in companies as discussed in section 1.10. Attempting to achieve a similar level of integration would be an effort in labour rather than innovation, which is not conducive to the scope of this project.

Majority of the analytical calculations are written as pre-defined functions stored in a database, shown in Appendix VII. This allows MIAMI the use of standardised functions that are retrieved from the database when necessary. As a result of a more streamlined script, individual sections of the software to be modified with limited effect on the intended function. An over-arching

script exists to hold all the independent functions together and connect each stage within MIAMI. However, individual functions can be removed from MIAMI and run independently if desired.

8.6.5 Rationale Behind Choosing the Python Language

While there are many choices for a programming language, python is the most straightforward. Although C based languages are better for more complex programming with faster running speed, python has all the necessary complexity for data processing.

8.7 Full *E. coli* Run

The MIAMI software is developed using simulated data from a previous manual run. The intricacies of replicating the decisions of a human user is rigorously fine-tuned. In this section, MIAMI is tested with processing a fresh set of data in a blind comparison test. For each stage, the data would be inspected, and optimal clones selected manually. After which, the raw data would be fed to the software. It is hoped that MIAMI would pick the same results as the manual selection. In turn, this would prove that the software does have the capabilities to intelligently evaluate strain performance.

8.7.1 Full *E. coli* Run Stage I

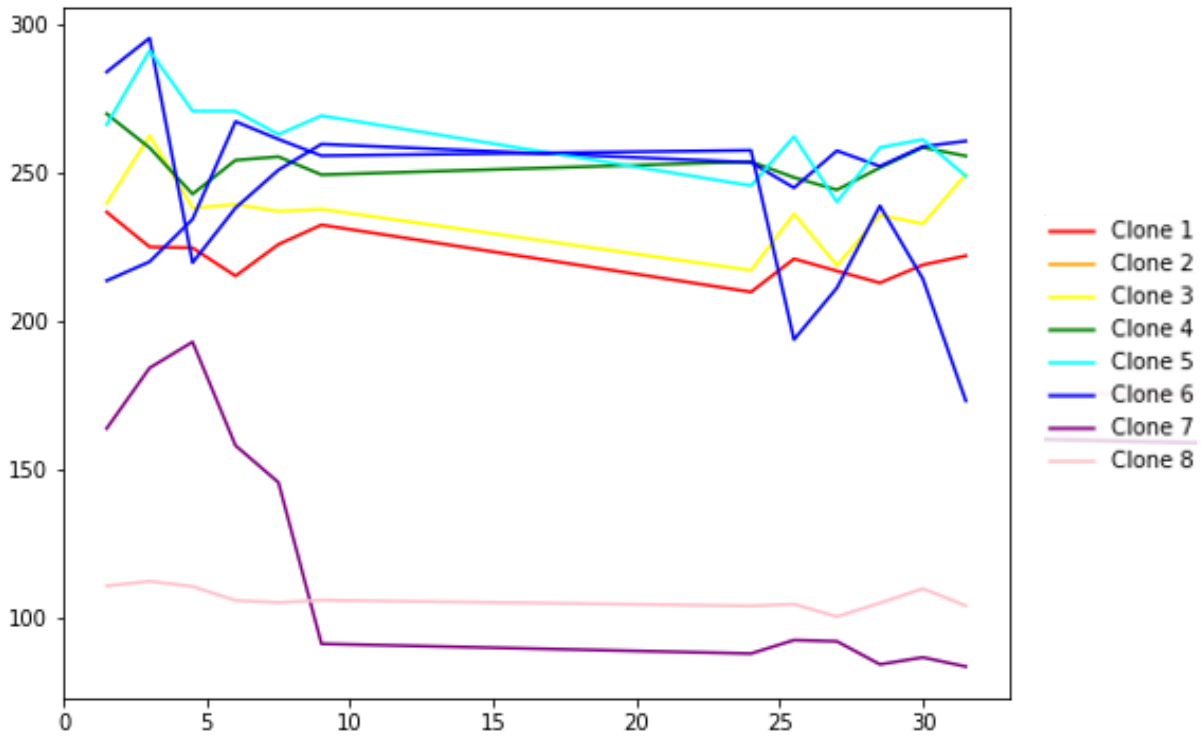
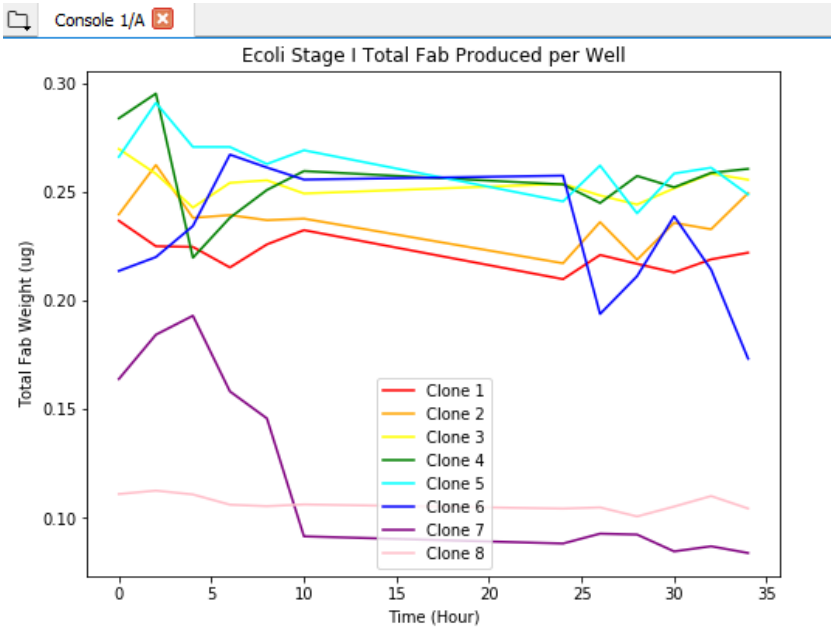


Figure 8.11: Graph produced by MIAMI showing total Fab' produced per 800 µL well for eight different *E. coli* clones in a full run.

Clone 7 and Clone 8 are noticeably not in contention for best yield, and arguably Clone 1 and Clone 2 both are underperforming in comparison to the remaining clones. Since the Fab' weight decreases over time as cell ruptures, more focus would be placed on the first 10 hours of the graph. Within this parameter, Clone 5 is the most stable and appears to have the highest yield and therefore is crowned the best performer. Unfortunately Clone 3, Clone 4, and Clone 6 appear to perform similarly, and is difficult to judge their rankings by eye.



	RANKING VALUE
CLONE 5	0.277
CLONE 4	0.266
CLONE 6	0.261
CLONE 3	0.259
CLONE 2	0.247
CLONE 1	0.229
CLONE 7	0.180
CLONE 8	0.111

```
Would you like to save the Fab Graph?No
The top three strains are:
1st: Clone 5
2nd: Clone 4
3rd: Clone 6
```

Figure 8.12: Console output showing the top three performing clones as evaluated by MIAMI (Left) and the arbitrary values of the clones used by MIAMI to perform the ranking (Right). The software ranks Clone 5, Clone 4 and Clone 6 as the highest performing clones in this run.

By looking at the ranking values, as expected, Clone 1, Clone 2, Clone 7 and Clone 8 were never in contention for best performance. Clone 5 out performs the rest by a significant margin. Clone 3, Clone 4 and Clone 6 holds very similar ranking values. However, since the MIAMI software can process the data with more precision, it appears Clone 4 and Clone 6 out ranked Clone 3 by a slight margin.

8.7.2 Full *E. coli* Run Stage II

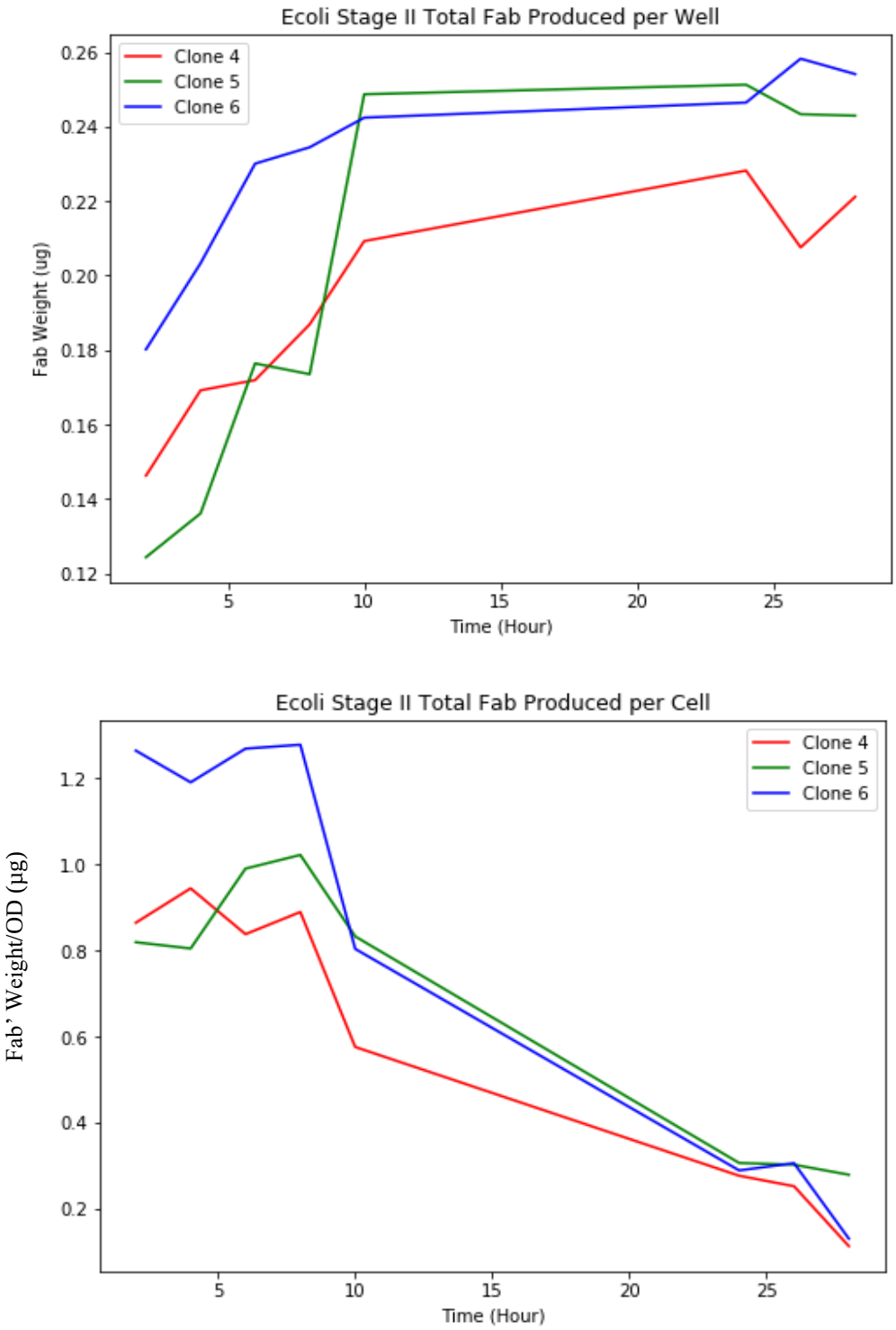
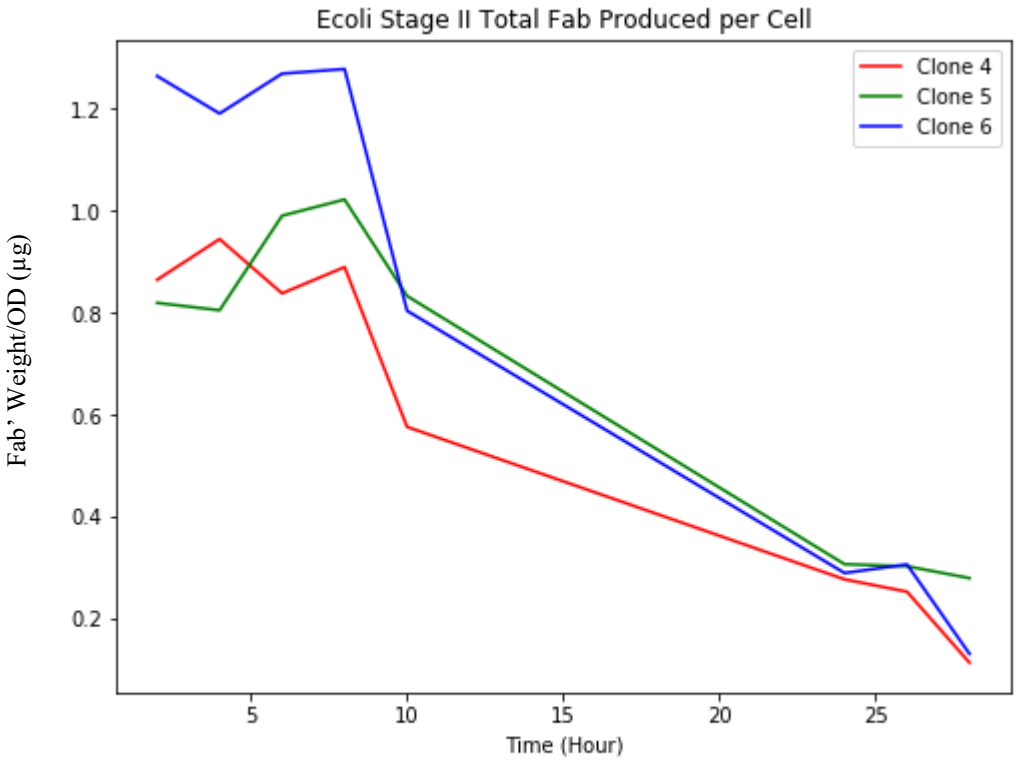


Figure 8.13: Total Fab' produced per 800 µL well (Top) and total Fab' produced per cell (Bottom) for the top three performing *E. coli* clones in this full run as chosen in Stage I.

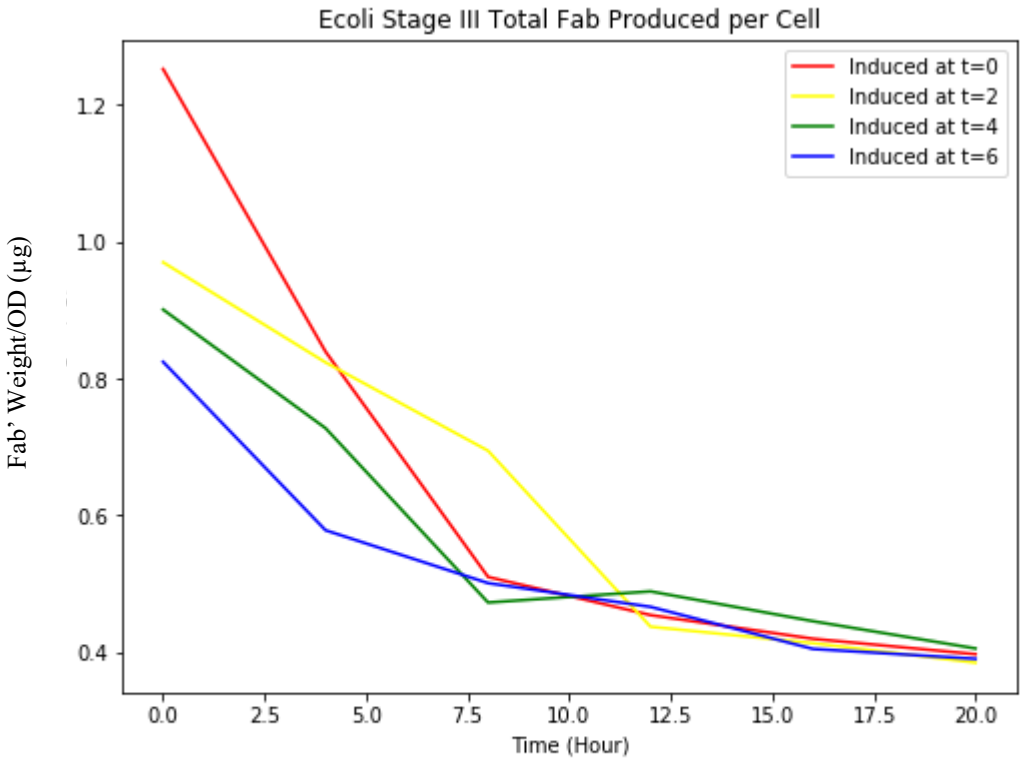
Looking at the top graph in Figure 8.7.2A, judging solely from total Fab' produced, Clone 5 appears to be the choice for the top performer. It has the highest productivity per well. However, taking into consideration the OD and observing the two bottom graph in Figure 8.7.2A, Clone 6 produced more Fab' per cell than the others by a significant margin. As a result, Clone 6 is the top performing clone. As can be seen in Figure 8.7.2B, the MIAMI software came to the same conclusion. This data set in conjunction with the data from Stage I illustrates why a secondary ranking stage is required. Even though Clone 6 was the lowest ranked of the top three resulting from Stage I. After more comprehensive cultivation sampling, it was in fact the best performing clone.




```
Would you like to proceed with default weighting? (Yes/No)Yes
Proceeding with default.
The top performing strain is: Clone 6
```

Figure 8.14: Console output showing the top performing *E. coli* clone in a full run as evaluated by MIAMI.

8.7.3 Full *E. coli* Run Stage III




```
Would you like to proceed with default weighting? (Yes/No)Yes
Proceeding with default.
The top performing strain is induced at t=2 at an OD of 0.218387166666666
```

Figure 8.15: Console output showing the optimal induction time for Clone 6 as evaluated by MIAMI. The optimal induction time is at the 2-hour mark, at an OD of approximately 0.218.

Examining the graph in Figure 8.7.3A, it appears induction after 2 hours of cultivation generate the highest yield. Although induction at inoculation had an initial spike of Fab' weight, the subsequent immediate loss of product indicates that it is not an optimal choice. The same evaluation is made by the MIAMI software.

In all three stages, the MIAMI software made decisions comparable to those that was made by a human user. This verifies that the code is able to make an intelligent assessment of the raw data provided to it.

8.8 Chapter Conclusion

MIAMI has been successfully developed for the appropriate ranking of multiple clones for *E. coli* and *P. pastoris*. It consists of two assay agents, one that processes raw HPLC data and one that processes raw flow cytometry data, and five intelligent ranking agents, three for *E. coli* and two for *P. pastoris*.

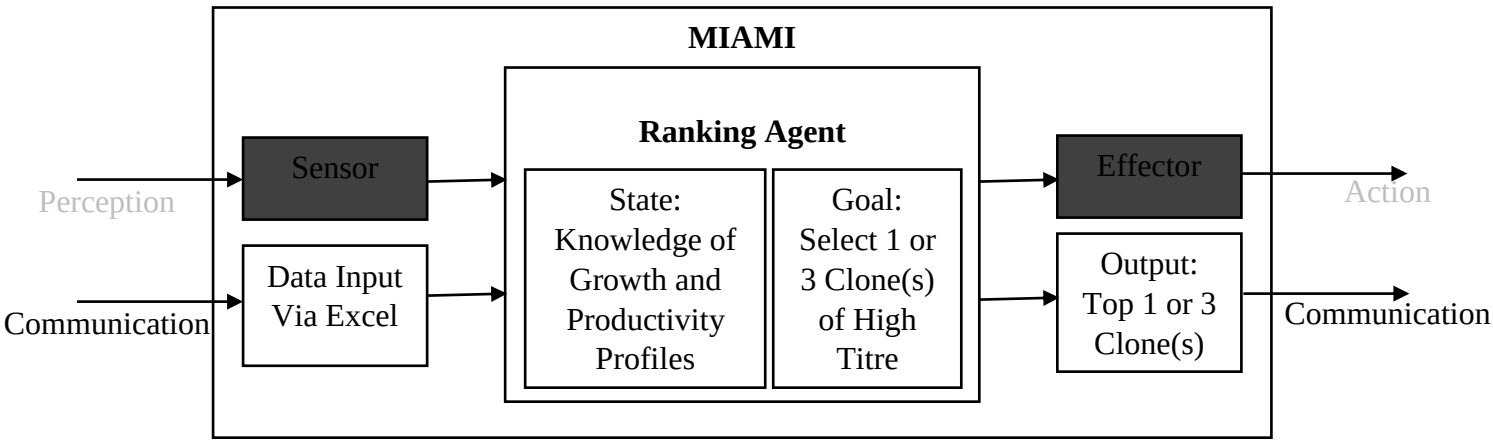


Figure 8.16: Diagram of the intelligent ranking agent in MIAMI. The ranking agent possesses knowledge of growth and productivity profiles and has the goal of selecting a top performing clone or clones. Data is communicated to the ranking agent and using its knowledge and set goal the ranking agent would make an appropriate selection of the optimal clone of clones. This decision output is then communicated to the user.

The diagram of a ranking agent is shown in Figure 8.8B. Raw data is communicated to MIAMI manually via an excel spreadsheet. The ranking agent is embedded with the knowledge of productivity profiles and has the ability to identify a stationary period of consistent product titre. The agent then calculates the average production titre during this period. Based upon the goal, the ranking agent would identify the clones with the highest achieved titre. This output is communicated to the user and can be used in next stage of clone selection.

At this point of MIAMI's development, it is unable to perceive data and actuate its decisions directly using the TECAN, HPLC or flow cytometer. As a result, the communication between the MIAMI software and the HT platforms are achieved manually through the aid of an operator. Through the manual input of raw data, MIAMI can process the data and rank the clones based upon two factors: highest stable yield and variability. Using a different set of *E. coli* clones, a blind comparison between human ranking and MIAMI ranking showed that both came to the same conclusion. This validates the success of MIAMI as a tool for bioprocess development

Chapter 9: Conclusion

The MIAMI software created for this project is a proof-of-concept of an automated intelligent upstream process development for heterologous protein production. In Chapter 3, an inverse methanol sensing gene network was transformed into *P. pastoris* cells and characterised in glycerol, methanol and sorbitol complex and minimal media cultures. In Chapter 5and Chapter 6, *E. coli* and *P. pastoris* strains expressing the same IP-free Fab' were created and characterised in complex media cultures. These two strains behaved as expected and the data from their characterisation were used to develop MIAMI in Chapter 8. This prototype software sets up a series of high-throughput experimentation that screens clones and rank them, with subsequent runs to further reaffirm individual clones' performance. The success of MIAMI is evaluated by the following criteria.

9.1 MIAMI's Ability to Collect the Necessary Data at a Small-Scale

Prior to the development of the intelligent agents and algorithms of MIAMI, it was important to ascertain if the three strains created can be cultivated in a HT small-scale environment. MIAMI would be redundant if the necessary data cannot be collected using the desired automated equipment. Thus, it was crucial that these strains can generate the intended Fab' and GFP signals at high enough quantities to be detected by the analytical equipment. For the development of a clone ranking agents, the difference between the performance of each clone must also be detectable at the intended small scale.

This was validated for the Fab' strains in Chapter 5 and Chapter 6, where a manual ranking of 8 clones was conducted where Fab' was detected for all clones. The best and worst clones of both strains had significantly different yields. The GFP strain was validated in Chapter 8, where the decay of GFP was evident when methanol was introduced to the small-scale cultivation.

9.2 Validation of MIAMI's Ranking at Large-Scale

As a predictor of each clone's performance, the entire automated platform and the ranking algorithm of MIAMI must translatable to a large-scale bioreactor. This was discussed in Chapter 7 when 3 clones for *E. coli* and *P. pastoris* achieved relative specific yield in both the small 800µL scale and 250mL bioreactor. Thus, demonstrating that using the automated small scale TECAN platform is a viable predictor of clones' performance. Although more validation work is required, and the algorithm might need some modification to account for factors that might only be present in larger-scale. The preliminary research in this project suggests that MIAMI is capable of screening clones for production in bioreactors.

9.3 Time Efficiency for Clone Screening Using MIAMI

From a process development viewpoint, for the MIAMI software to have industrial application, it must perceptibly reduce the amount of time required to process the data from clone screening. In the context of this project, only 8 clones were ever simultaneously evaluated. While it might have taken a human researcher approximately thirty minutes to an hour to amass the data and process the results for evaluation, it would only take the software a few seconds.

While the time reduction to screen for 8 clones does not immediately appear to have a significant impact, once the number of clones screened is scaled up the time saved will increase exponentially. The software accelerates not just purely in terms of time required for data processing, but also the time spent evaluating the performance of each clone. It is easy to distinguish the performance of just 8 clones, however, as the number of clones increases, the small difference in performance would be harder to evaluate by the human eye.

Table 9.1: Estimation of the time required to evaluate different number of clones manually and using MIAMI. Linearly scaling of time saved using MIAMI shows an 87% decrease in clone analysis.

NUMBER OF CLONES	MANUAL EVALUATION (MINUTE)	MIAMI EVALUATION (MINUTE)	TIME SAVED (MINUTE)
8	30	5	25
50	187.5	31.25	156.25
100	375	62.5	312.5
200	750	125	625
500	1875	312.5	1562.5

Using Stage I of the MIAMI software was example. The data set took about half an hour to evaluate manually. This estimated is based upon the time taken to evaluate the Stage I *E. coli* and *P. pastoris* data sets in Chapter 5 and Chapter 6 respectively. This time includes, processing raw data, generating the graph, and assessing the resulting graph to select for three optimal clones. When using the MIAMI software in Chapter 8, this entire process took less than 5 minutes. Assuming a linear increase in time taken, the amount for both manual and MIAMI screening in Stage I the time saved increases dramatically. When screening up 100 clones, the amount of time saved in just one stage would be approximately 5 hours. This is a modest estimation operating under the assumption that no human errors would be made when processing this large amount of data manually, and not taking into account the amount of time saved using automated equipment.

9.4 Validation of Intelligent Ranking in MIAMI

The intelligent aspect of MIAMI would be reflected in the appropriateness of its clone ranking. While it would be impossible to completely capture the entire processes with the personal preference of each individual user, the algorithm in MIAMI is set up to look at the clone's data set objectively. Considering two significant product criteria, yield and stability, MIAMI tries to reflect the priorities of a researcher when performing clone selection. When conducting a blind trial to validate the intelligent ranking algorithm, MIAMI selected the same top performing clones that a researcher had.

To facilitate a wider range of users, flexibility has been incorporated into the MIAMI software itself. A variable weighting can be placed on the two factors. Each user can apply their own standards to the ranking process, therefore is able to customise the results to suit their product.

Chapter 10: Future Work

10.1 Incorporating a Clone Screening Process for CHO in MIAMI

The obvious immediate step is to expand the MIAMI software to include a ranking system for CHO strains. Clonal selection is arguably a more significant aspect for mammalian cells. Not only would there likely be more variability between each clone depending on how the plasmid DNA is incorporated into the cell genome, the screening would also have to be able to differentiate between stable and transient transfected cell lines. Establishing the MIAMI software for CHO cells would have more significant impact than compared to *E. coli* and *P. pastoris*. Since not only would the software save the researcher time when screening multiple clones, it would also lessen media consumption. A concern that affects mammalian cells considerably more due to the high cost of its media.

Attempts at high-throughput systems for mammalian cells have been reported to be successful in small-scale microtiter plates, working for volumes as low as 800µL (Bareither, et al., 2013). Thus, developing and integrating a screening process for CHO hosts would be straightforward. The same set up in the automated TECAN platform should not require many major changes. The overall process would just be over a longer duration of time, to account for the slower growth rate of mammalian cells.

Establishing the same Fab' across *E. coli*, *P. pastoris* and CHO will provide data for enhanced learning for host cell comparison.

10.2 Upstream Process Development: Media Composition and Culture Conditions

Continuing from clone selection, the subsequent steps in optimising an upstream process would involve media components and culture conditions optimisation. Defined media can be tailored to each specific strain to not only increase specific yield but also ensure a consistent titre, making it far more beneficial than the generic complex media. Characterising media components and adjusting for the appropriate amount would require a large amount of time if done using the antiquated one factor at a time technique. Media optimisation using mathematical and statistical techniques has become wildly used as a more effective and economical technique (Singh, et al., 2017).

Media optimisation and culture conditions optimisation using DOE technique is common in recent developments. DOE is a series of experimentation what are strategically planned and executed to cover a large design space while using a limited number of experiments. Effects of multiple factors are compared simultaneously, and interactive between effects observed. By applying this existing method to in a small-scale allows for more parameters to be explored and high-throughput methodology supporting parallel experimentations.

Culture conditions such as temperature and pH, would be easy to scale up once optimised. However, agitation, and by extension dissolved oxygen level, would be harder to scale-up correctly. More research and scale-up studies would have to be conducted to be fully able to optimise using scaled-down methods.

10.3 Integrating MIAMI with TECAN and Analytical Equipment

MIAMI currently does not have the ability to communicate directly with HT platforms such as TECAN, HPLC or flow cytometer. Thus, it is unable to retrieve data automatically and is currently receiving data input manually. Integrating MIAMI with these equipment creates a more streamlined process, reducing required contact time and further increase the time efficiency. With a fully integrated system, MIAMI would be able to perform all three stages of clone selection with little interaction with an operator.

10.4 Integration of a Data Bank to Facilitate Machine Learning Capacity

Improvements in data storage capacity facilities the use of large data banks to determine trends in performance for similar strains. This is a feature that would greatly benefit the media optimisation steps. Because media optimisation was not coded in the prototype MIAMI software, there are currently no learning capability. As an additional feature, MIAMI can be easily integrated with a code to store the results from all experimental runs in a historic data bank. When a large enough bank of data is established, the data can be used to predict how similar strains might be affected by varying media components and culture conditions.

10.5 Feedback from Downstream

Optimising an upstream process would naturally prioritise high titres. It is logical to assume that more cells generated would translate to more specific yield. While this might be true for upstream processing, higher titres would often correlate with more purification burden during downstream processing due to impurities. A feedback loop from downstream processing can help guild the upstream process development, to create conditions that are ideal for the entire process. Selecting a clone with fewer impurities at the cost of a lower titre could be advantageous. By alleviating the common rate limiting purification step in downstream processing, the overall process can be managed in a much more efficient manner.

References

Adler, P. S., Mandelbaum, A., Nguyen, V. & Schewerer, E., 1995. From Project to Process Management: An Empirically-based Framework for Analyzing Product Development Time. *Management Science,* 41(3), pp. 458-484.

Amanullah, A. et al., 2009. Novel Micro-Bioreactor High Throughput Technology for Cell Culture Process Development: Reproducibility and Scalability Assessment of Fed-Batch CHO Cultures. *Biotechnology and Bioengineering,* pp. 57-67.

Baltz, R. H., Demain, A. L. & Davies, J. E., 2010. *Manual of Industrial Microbiology and Biotechnology.* 3rd ed. Washington: American Society for Microbiology Press.

Bareither, R. et al., 2013. Automated Disposable Small scale reactor for High Throughput Bioprocess Development: A Proof of Concept Study. *Biotechnology and Bioengineering,* 110(12), pp. 3126-3138.

Berrios, J. et al., 2017. A comparative study of glycerol and sorbitol as co-substrates in methanol-induced cultures of Pichia pastoris: temperature effect and scale-up simulation. *Journal of Industrial Microbiology and Biotechnology,* 44(3), pp. 407-411.

Bessette, P. H. et al., 2001. Effect of Sequences of the Active-Site Dipeptides of DsbA and DsbC on In Vivo Folding of Multidisulfide Proteins in Escherichia coli. *Journal of Bacteriology,* pp. 980-988.

Bhambure, R., Kumar, K. & Rathore, A. S., 2011. High-throughput process development for biopharmaceutical drug substances. *Trends in Biotechnology,* 29(3), pp. 127-135.

Birch, J. R. & Racher, A. J., 2006. Antibody production. *Advance Drug Delivery Reviews,* 58(5-6), pp. 671-685.

Bos, A. B. et al., 2015. Optimization and Automation of an End-to-End High Throughput Microscale Transient Protein Production Process. *Biotechnology and Bioengineering,* pp. 112: 1832-1842.

Carrier, T. et al., 2010. High-Throughput Technologies in Bioprocess Development. In: M. C. Flickinger, ed. *Encyclopedia of Industrial Biotechnology: Bioprocess, Bioseparation, and Cell Technology.* s.l.:John wiley & Son, pp. 1-31.

Čepl, J., Blahůšková, A., Neubauer, Z. & Markoš, A., 2016. Variations and heredity in bacterial colonies. *Communicative & Integrative Biology,* 9(6), pp. 1-11.

Chhina, M., 2013. *Overview of Biological Products,* U.S: FDA.

Costa, A. R. et al., 2010. Guidelines to cell engineering for monoclonal antibody production. *European Journal of Pharmaceutics and Biopharmaceutics,* 74(2), pp. 127-138.

de Boer, H. A., Comstock, L. J. & Vasser, M., 1983. The tac promoter: a functional hybrid derived from the trp and lac promoters. *Proceedings of the National Academy of Sciences of the United States of America,* 80(1), pp. 21-25.

de Hoon, M. J., Makita, Y., Nakai, K. & Miyano, S., 2005. Prediction of transcriptional terminators in Bacillus subtilis and related species. *PLoS Computational Biology,* 1(3), pp. 0212-0221.

Feng , S. C. & Song, E. Y., 2002. *Preliminary Design and Manufacturing Planning Integration Using Intelligent Agents.* Rio de Janeiro, IEEE.

Frenzel, A., Hust, M. & Schirrmann, T., 2013. Expression of recombinant antibodies. *Frontiers in Immunology,* Volume 4, pp. 1-20.

Frenzel, A., Hust, M. & Schirrmann, T., 2013. Expression of Recombinant Antibodies. *Frontiers in Immulogy,* pp. 217: 1-12.

Gao, Y. et al., 2009. Application of Agent-Based System for Bioprocess Description and Process Improvement. *Biotechnology Progress*, pp. 706-716.

Gawlitzek, M., Estacio, M., Fürch, T. & Kiss, R., 2009. Identification of Cell Culture Conditions to Control N-Glycoproteins Expressed in CHO Cells. *Biotechnology and Bioengineering*, 103(6), pp. 1164-1175.

Genesereth, M. R. & Ketchpel, S. P., 1994. Software Agents. *Communications*, pp. 48-53.

Glassey, J., Montague, G. & Willis, M., 2000. *Bioprocesses: A Challenge for the Application of Artificial Intelligence.* Newcastle upon Tyne, IASTED Applied Informatics AI'2000.

Gomez, N. et al., 2010. Effect of temperature, pH, dissolved oxygen, and hydrolysate on the formation of triple light chain antibodies in cell culture. *Cell Culture and Tissue Engineering*, 26(5), pp. 1438-1445.

Gronemeyer, P., Ditz, R. & Strube, J., 2014. Trends in Upstream and Downstream Process Development for Antibody Manufacturing. *Bioengineering*, pp. 188-212.

Heads, J. T. et al., 2012. Relative Stabilities of IgG1 and IgG4 Fab Domains: Influence of the Light-Heavy Interchain Disulfide Bond Architecture. *Protein Science*, pp. 1315-1322.

Holliger, P. & Hudson, P. J., 2005. Engineered antibody fragments and the rise of single domains. *Nature Biotechnology*, Volume 23, pp. 1126-1136.

Holliger, P. & Hudson, P. J., 2005. Engineered antibody fragments and the rise of single domains. *Nature Biotechnology*, 23(9), pp. 1126-1136.

Hsu, C. C., Thomas, O. R. T. & Overton, T. W., 2015. Periplasmic Expression in and Release of Fab Fragments from Escherichia coli Using Stress Minimization. *Journal of Chemical Technology and Biotechnology.*

Hsu, C.-C., Thomas, O. & Overton, T., 2016. Periplasmic expression in and release of Fab fragments from Escherichia coli using stress minimisation. *Journal of Chemical Technology and Biotechnology*, 91(3), pp. 815-822.

Hsu, W. T., Aulakh, R. P. S., Traul, D. L. & Yuk, I. H., 2012. Advance Microscale Bioreactor System: A Representative Scale-Down Model for Bench-Top Bioreactors. *Cytotechnology*, pp. 667-678.

Hubbuch, J., 2012. Editorial: High-throughput process development. *Biotechnology Journal*, 7(10), p. 1185.

Hu, Y. et al., 2016. Optimization of Saccharomyces boulardii production in solid-state fermentation. *Biotechnology & Biotechnological Equipment*, 30(1), pp. 173-179.

Ishii, Y. et al., 2014. Efficient Folding/Assembly in Chinese Hamster Ovary Cells is Critical for High Quality (Low Aggregate Content) of Secreted Transtuzumab as well as for High Production: Stepwise Multivariate Regression Analysis. *Journal of Bioscience and Bioengineering*, pp. 118: 223-230.

Jain, E. & Kumar, A., 2008. Upstream processes in antibody production: Evaluation of critical parameters. *Biotechnology Advances*, pp. 46-72.

Jin, M. et al., 2009. Profiling of Host Cell Proteins by Two-Dimensional Difference Gel Electrophoresis (2D-DIGE): Implications for Downstream Process Development. *Biotechnology and Bioengineering*, 105(2), pp. 306-316.

Jonakin, K., 2016. The Future of Biologics Drug Development is Today. *Sciex*, 18 March.

Jordon, M. et al., 2013. Cell culture medium improvement by rigorous shuffling of components using media blending. *Cytotechnology*, Volume 65, pp. 31-40.

Kunert, R., Gach, J. & Katinger, H., 2008. Expression of a Fab Fragment in CHO and Pichia pastoris. *BioProcess International,* Volume June 2008, pp. 34-40.

Kunert, R., Gach, J. & Katinger, H., 2008. Expression of a Fab Frament in CHO and Pichia pastoris. *Bioprocess International ,* pp. 34-40.

Kunert, R. & Reinhart, D., 2016. Advances in recombinant antibody manufacturing. *Applied Microbiology and Biotechnology,* Volume 100, pp. 3451-3461.

Kurihara, T. et al., 1992. *ANTI-HBs ANTIBODY GENE AND EXPRESSION PLASMID THEREOF*. Japan, Patent No. WO 1993020205 A1.

Lane, J., 2018. The Top 10 advances in renewable butanol: what's speeding up, where are the slow-downs?. *Biofuels Digest*, 2 8.

Langer, E. S. & Rader, R. A., 2014. Single-use technologies in biopharmaceutical manufacturing: A 10-year review of trends and the future. *Engineering in Life Sciences,* 14(3), pp. 238-243.

Lattermann, C. & Buchs, J., 2015. Microscale and Miniscale Cultivationand Screening. *Current Opinion in Biotechnology,* pp. 35: 1-6.

Li, F. et al., 2010. Cell culture processes for monoclonal antibody production. *Pharmaceutical Sciences Encyclopaedia: Drugs, Discover, Development, and Manufacturing,* 2(5), pp. 466-479.

Lim, J. A. C. et al., 2010. Modeling Bioprocess Cost. *BioProcess International,* Volume 2010, pp. 62-69.

Lim, J. A. C. et al., 2010. Modeling Bioprocess Cost: Process Economic Benefits of Expression Technology Based on Pseudomonas fluorescens. *BioProcess International,* pp. 62-70.

Liu, H. & May, K., 2012. Disulfide Bond Structures of IgG Molecules. *Landes Bioscience*, pp. 17-23.

Li, Z. et al., 2014. Novel Insight into the Secretory Expression of Recombinant Enzymes in Escherichia Coli. *Process Biochemistry*, pp. 49: 599-603.

Looser, V. et al., 2015. Cultivation strategies to enhance productivity of Pichia pastoris: A review. *Biotechnology Advances,* 33(6), pp. 1177-1193.

Meyer, H.-P. & Schmidhalter, D. R., 2012. Microbial Expressioin Systems and Manufacturing from a Market and Economic Perspective. In: E. C. Agbo, ed. *Innovations in Biotechnology.* London: InTech, pp. 211-250.

Meystel, A. M. & Messina, E. R., 2000. *Measure the Performance and Intelligence of Systems: Proceedings of the 2000 PerMIS Workshop.* Gaithersburg, NIST Special Publication 970.

Morris, C. & Segal, J., 2009. *Some challenges facing scientific software developers: The case of molecular biology.* Oxford, IEEE e-Science.

Nienow, A. W. et al., 2013. THe Physical Characterisation of a Microscale Parallel Bioreactor Platform with an Industrial CHO Cell Line Expressing an IgG4. *Biochemical Engineering Journal*, pp. 25-36.

Nor, N. M. et al., 2017. Comparative analyses on medium optimization using one-factor-at-a-time, response surface methodology, and artificial neural network for lysine-methionine biosynthesis by Pediococcus pentosaceus RF-1. *Biotechnology & Biotechnological Equipment,* 31(5), pp. 935-947.

Ozturk, S. S. & Palsson, B. O., 1990. Effects of dissolved oxygen on hybridoma cell growth, metabolism, and antibody production kinetics in continuous culture. *Biotechnology Progress,* 6(6), pp. 437-446.

Petit, T. & Petit, L., 2016. *Optimizing Molecular Cloning of Multiple Plasmids.* Buenos Aires, IJCAI/AAAI.

Pisano, G., 1994. Knowledge, Intergration, and the Locus of Learning: An Empirical Analysis of Process Development. *Strategic Management Journal,* Volume 15, pp. 85-100.

Rameez, S., Mostafa, S. S., Miller, C. & Shukla, A. A., 2014. High-Throughput Miniaturized Bioreactors for Cell Culture Process Development: Reproducibility, Scalability, and Control. *Biotechnology Process,* pp. 718-727.

Rosso, L., Lobry, J. R., Bajard, S. & Flandrois, J. P., 1995. Convenient Model To Describe the Combined Effects of Temperature and pH on Microbial Growth. *Applied and Environmental Microbiology,* 61(2), pp. 610-616.

Russell, S. J. & Peter, N., 2003. *Artificial Intelligence: A Modern Approach.* 2nd ed. Upper Saddle River: New Jersey: Prentice Hall.

Shen, B., Hohmann, S., Jensen, R. G. & Bohnert, A. H., 1999. Roles of sugar alcohols in osmotic stress adaptation. Replacement of glycerol by mannitol and sorbitol in yeast.. *Plant Physiology,* 121(1), pp. 45-52.

Shioya, S., Shimizu, K. & Yoshida, T., 1999. Knowledge-Based Design and Operation of Bioprocess Systems. *Journal of Bioscience and Bioengineering,* 87(3), pp. 261-266.

Shu, J. et al., 2006. Siliencing of bidirectional promoters by DNA methylation in tumorigenesis. *Cancer Research,* 66(10), pp. 5077-5084.

Shukla, A. & Thömmes, J., 2010. Recent advances in large-scale production of monoclonal antibodies and related proteins. *Trends in Biotechnology,* 28(5), pp. 253-261.

Singh, V. et al., 2017. Strategies for CultivationMedium Optimization: An In-Depth Review. *Frontiers in Microbiology,* 7(2087), pp. 1-16.

Siurkus, J. & Neubauer, P., 2011. Heterologous Production of Active Ribonuclease Inhibitor in Escherichia coli by Redox State Control and Chaperonin Coexpression. *Microbial Cell Factories,* pp. 10: 65-76.

Šiurkus, J. & Neubauer, P., 2011. Reducing conditions are the key for efficient production of active ribonuclease inhibitor in Escherichia coli. *Microbial Cell Factories,* 10(31), pp. 1-15.

Sommerfeld, S. & Strube, J., 2005. Challenges in biotechnology production-generic processes and process optimization for monoclonal antibodies. *Chemical Engineering and Processing: Process Intensification,* 44(10), pp. 1123-1137.

Spiro, R. G., 2002. Protein Glycosylation: Nature, Distribution, Enzymatic Formation, and Disease Implications of Glycopeptide Bonds. *Glycobiology,* pp. 43R-56R.

Swartz, J. R., 2001. Advances in Escherichia coli production of therapeutic proteins. *Current Opinion in Biotechnology,* 12(2), pp. 195-201.

Takai, D. & Jones, P. A., 2004. Analyses of Intergenic Distance in the Human Genome. *Molecular Biology and Evolution,* 21(3), pp. 463-467.

Tatikonka, M. & Montoya-Weiss, M. M., 2001. Integrating Operations and Marketing Perspectives of Product Innovation: The Influence of Organizational Process Factors and capabilities on Development Performance. *Management Science,* 47(1), pp. 151-172.

TreDenick, T., 2018. The Evolution of Therapeutic Monoclonal Antibodies. *Pharmaceutical*, 14 August.

Ukkonen, K., Veijola, J., Vasala, A. & Neubauer, P., 2013. Effect of culture medium, host strain and oxygen transfer on recombinant Fab antibody fragment yield and leakage to medium in shaken *E. coli* cultures. *Microbial Cell Factories,* 12(73), pp. 1-14.

Ukkonen, K., Veijola, J., Vasala, A. & Neubauer, P., 2013. Effect of Culture Medium, Host Strain and Oxygen Transfer on Recombinant Fab Antibody Fragment Yield and Leakage to Medium in Shaken *E. coli* Cultures. *Microbial Cell Factories,* pp. 12: 73-87.

Valliere-Douglass, J. F. et al., 2010. Gluamine-linked and Non-consensus Asparagine-linked Oligosaccharides Present in Human Recombinant Antibodies Define Novel Protein Glycosylation Motif. *The Journal of Biological Chemistry,* pp. 16012-16022.

Verma, R., Boleti, E. & George, A. J. T., 1998. Antibody Engineering: Comparison of Bacterial, Yeast, Insect and mammalian Expression Systems. *Journal of Immunological Methods,* pp. 165-181.

Vermeer, A. W. P. & Norde, W., 2000. The Thermal Stability of Immunoglobulin: Unfolding and Aggregation of a Multi-Domain Protein. *Biophysical Journal,* pp. 394-404.

Wei, Y.-C.et al., 2017. Biotransformation of β-hydroxypyruvate and glycolaldehyde to l-erythrulose by Pichia pastoris strain GS115 overexpressing native transketolase. *Biotechnology Progress,* 34(1), pp. 99-106.

Werner, R. G., Noe, W., Kopp, K. & Schluter, M., 1998. Appropriate mammalian expression systems for biopharmaceuticals. *Drug Research,* 48(8), pp. 870-880.

Wu, T. & Zhou, Y., 2014. An Intelligent Automation Platform for Rapid Bioprocess Design. *Journal of Laboratory Automation,* pp. 19: 381-393.

Yang, S.-T. & Liu, X., 2013. Cell culture processes for biologics manufacturing: recent developments and trends. *Pharmaceutical Bioprocessing,* 1(2), pp. 133-136.

Yin, J. C., Li, G. X., Ren, X. F. & Herrler, G., 2007. Select What You Need: A Comparative Evaluation of the Advantages and Limitations of Frequently Used Expression Systems for Foreign Genes. *Journal of Biotechnology,* pp. 335-347.

Yin, J., Li, G., Ren, X. & Herrler, G., 2007. Select what you need: A comparative evaluation of the advantages and limitations of frequently used expression systems for foreign genes. *Journal of Biotechnology*, 127(3), pp. 335-347.

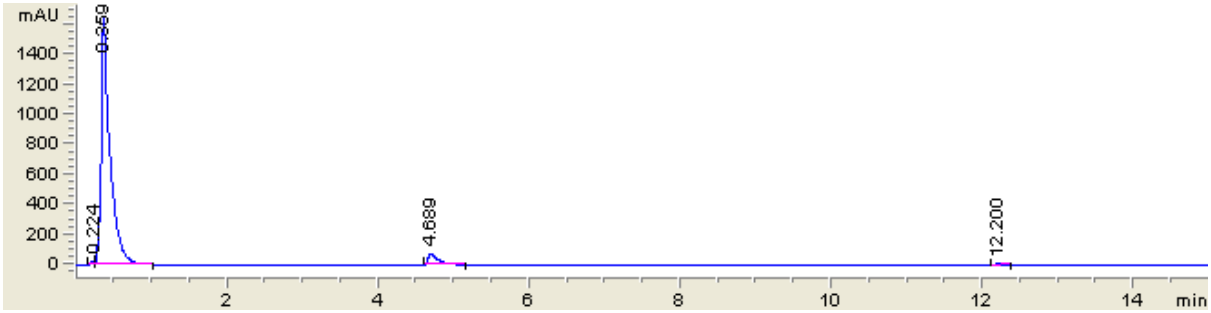
Yu, P., Zhu, Q., Chen, K. & Lv, X., 2015. Improving the secretory production of the heterologous protein in Pichia pastoris by focusing on protein folding. *Applied Biochemical Biotechnology*, 175(1), pp. 535-548.

Zhou, Z. et al., 2016. Codon Usage in an Important Determinant of Gene Expression Levels Largely Through Its Effects on Transcription. *Proceedings of the National Academy of Sciences of the United States of America*, 113(41), pp. E6117-E6125.

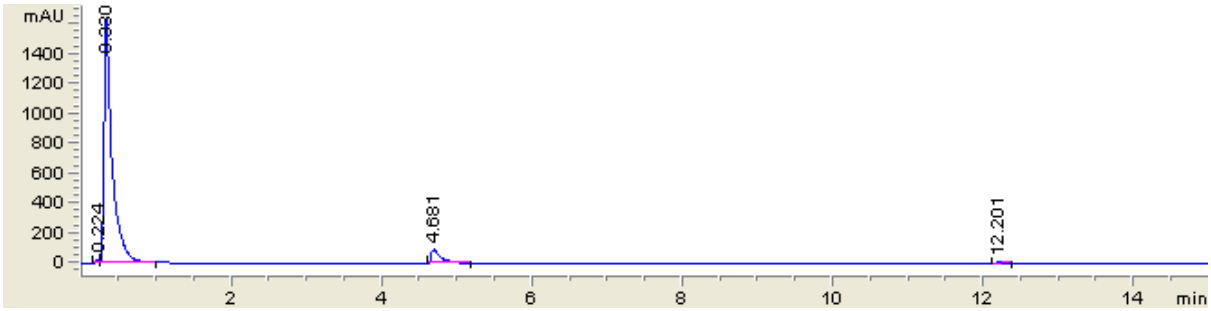
Appendix

Appendix I: Full Profiles of *E. coli* HPLC Results

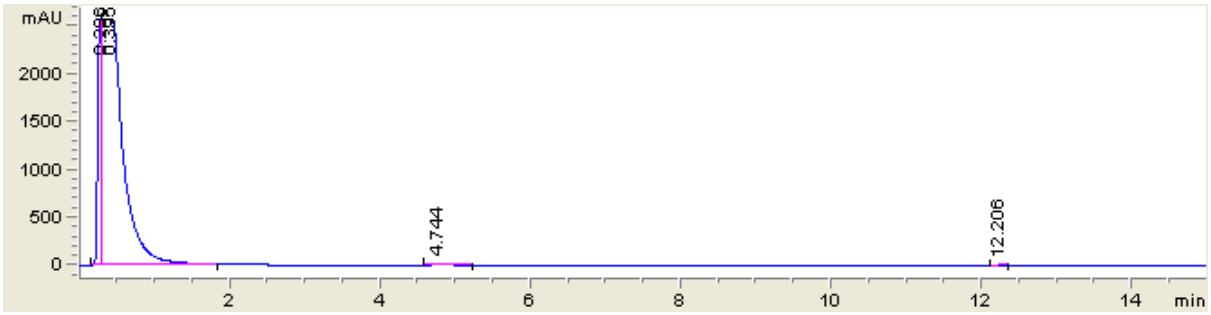
Uninduced TAHT66



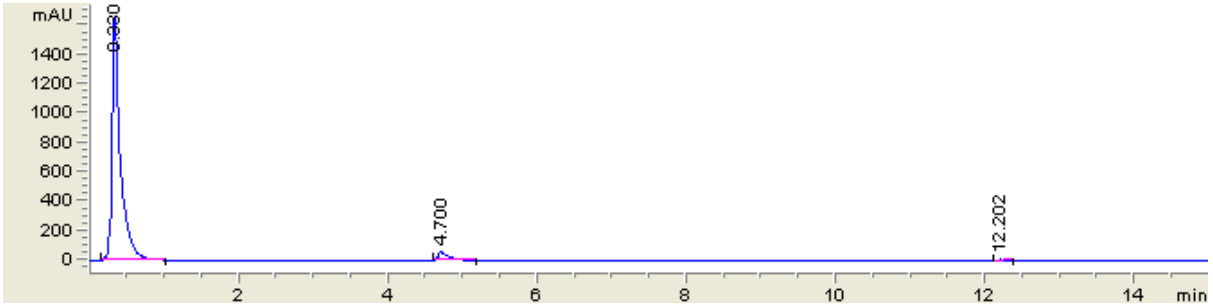
Induced QT-TAHT66



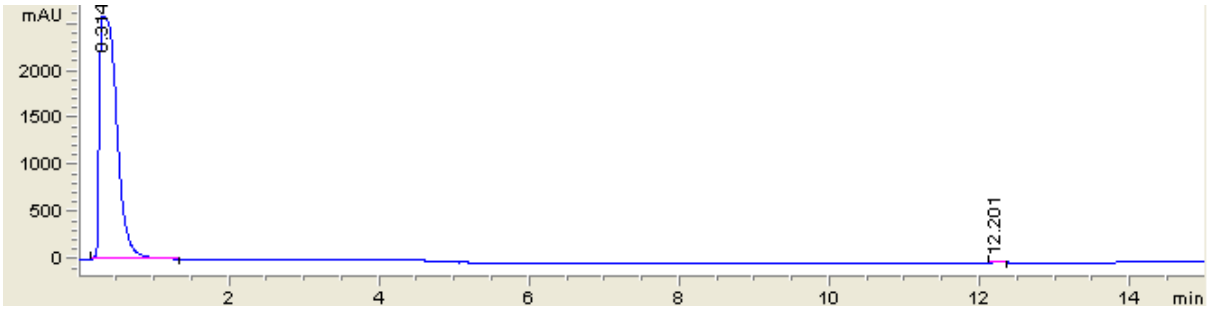
Uninduced QT-WAHT66



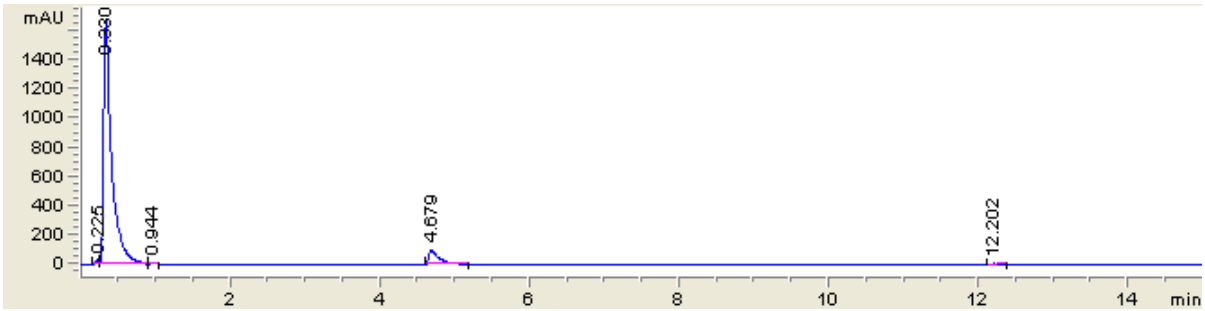
Induced QT-WAHT66



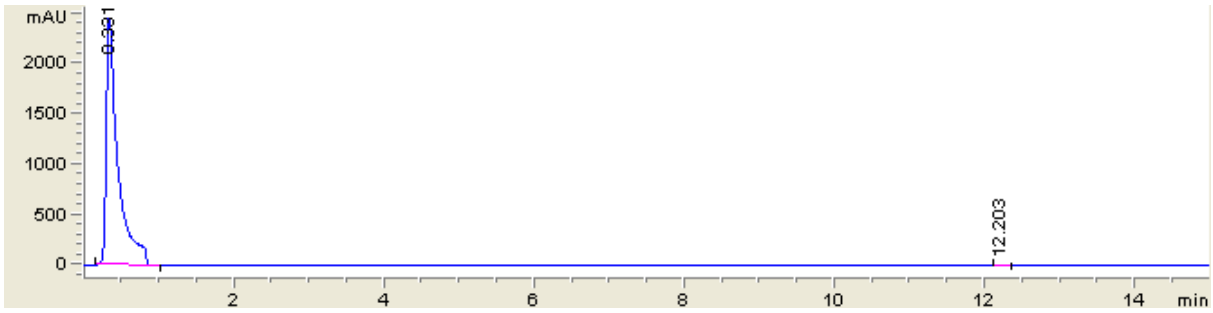
Uninduced UCB



Induced UCB

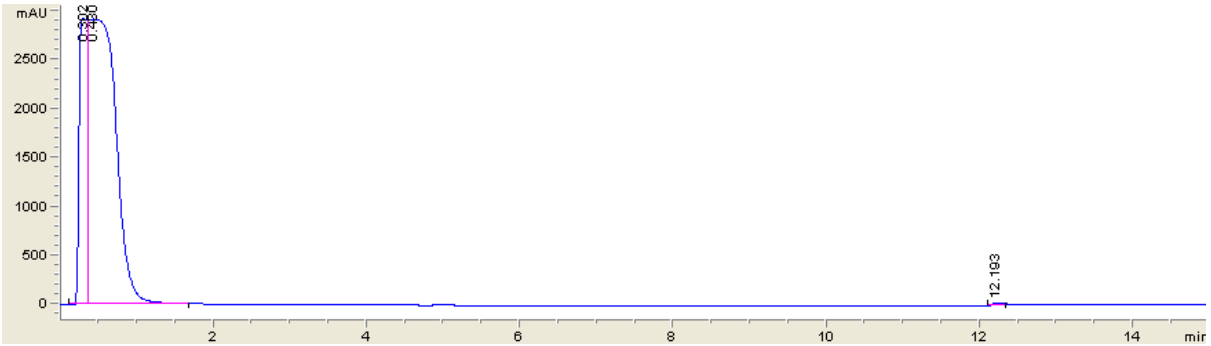


Induced W3110

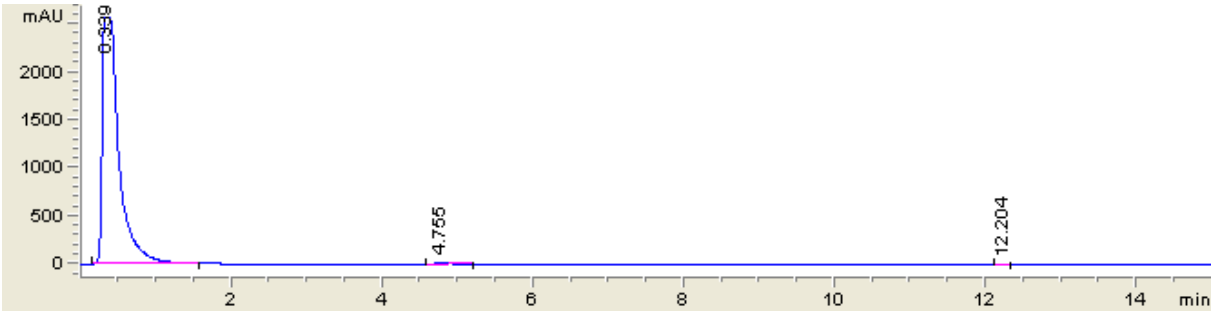


Appendix II: Full Profiles of *P. pasotris* HPLC Results

Induced GS115



Induced QT-GAHT



Appendix III: Code for Calculating the Constants for Least Square Fit Decay Equation

```
def Least_Square_Fit_A(x,y,up_lim,low_lim):
    i= low_lim - 1
    n= up_lim -1
    sum_xxy = x[int(i)] * x[int(i)] * y[int(i)]
    sum_ylny = y[int(i)] * np.log(y[int(i)])
    sum_xy = x[int(i)]* y[int(i)]
    sum_xylny = x[int(i)]* y[int(i)] * np.log(y[int(i)])
    sum_y = y[int(i)]
    i_1 = int(i) + 1
    while n >= i_1:
        sum_xxy = sum_xxy + x[int(i_1)] * x[int(i_1)] * y[int(i_1)]
        sum_ylny = sum_ylny + y[int(i_1)] * np.log(y[int(i_1)])
        sum_xy = sum_xy + x[int(i_1)]* y[int(i_1)]
        sum_xylny = sum_xylny + x[int(i_1)]* y[int(i_1)] * np.log(y[int(i_1)])
        sum_y = sum_y + y[int(i_1)]
        i_1 = i_1 + 1
    a = (sum_xxy*sum_ylny - sum_xy*sum_xylny) / (sum_y * sum_xxy - sum_xy * sum_xy)
    true_A = np.exp(a)
    return true_A
```

```
def Least_Square_Fit_B(x,y,up_lim,low_lim):
    i= low_lim - 1
    n= up_lim -1
    sum_xxy = x[int(i)] * x[int(i)] * y[int(i)]
    sum_ylny = y[int(i)] * np.log(y[int(i)])
    sum_xy = x[int(i)]* y[int(i)]
    sum_xylny = x[int(i)]* y[int(i)] * np.log(y[int(i)])
    sum_y = y[int(i)]
    i_1 = int(i) + 1
    while n >= i_1:
        sum_xxy = sum_xxy + x[int(i_1)] * x[int(i_1)] * y[int(i_1)]
        sum_ylny = sum_ylny + y[int(i_1)] * np.log(y[int(i_1)])
        sum_xy = sum_xy + x[int(i_1)]* y[int(i_1)]
        sum_xylny = sum_xylny + x[int(i_1)]* y[int(i_1)] * np.log(y[int(i_1)])
        sum_y = sum_y + y[int(i_1)]
        i_1 = i_1 + 1
    b = (sum_y*sum_xylny - sum_xy*sum_ylny) / (sum_y*sum_xxy - sum_xy*sum_xy)
    return b
```

Appendix IV: Code for Stage I *E. coli* Clones Ranking

```
import pandas

import numpy as np

import matplotlib.pyplot as plt

import Defun as dfc

from sys import exit



print('Please enter values into DATA INPUT ECOLI STAGEI.xlsx.')

Values_Entered = input("Enter Yes when done")

if Values_Entered == "Yes":

    print("Thank You!")

else:

    exit()



df = pandas.read_excel('DATA INPUT ECOLI STAGE

I.xlsx',sheet_name=0,index_col=None)

ODX = df['Time'].tolist()

OD1A = df['Clone 1'].tolist()

OD2A = df['Clone 2'].tolist()

OD3A = df['Clone 3'].tolist()
```

```
OD4A = df['Clone 4'].tolist()

OD5A = df['Clone 5'].tolist()

OD6A = df['Clone 6'].tolist()

OD7A = df['Clone 7'].tolist()

OD8A = df['Clone 8'].tolist()


df = pandas.read_excel('DATA INPUT ECOLI STAGE
I.xlsx',sheet_name=1,index_col=None)

OD1B = df['Clone 1'].tolist()

OD2B = df['Clone 2'].tolist()

OD3B = df['Clone 3'].tolist()

OD4B = df['Clone 4'].tolist()

OD5B = df['Clone 5'].tolist()

OD6B = df['Clone 6'].tolist()

OD7B = df['Clone 7'].tolist()

OD8B = df['Clone 8'].tolist()


OD1B_1 = dfc.List_X(OD1B,4)

OD2B_1 = dfc.List_X(OD2B,4)

OD3B_1 = dfc.List_X(OD3B,4)
```

```
OD4B_1 = dfc.List_X(OD4B,4)

OD5B_1 = dfc.List_X(OD5B,4)

OD6B_1 = dfc.List_X(OD6B,4)

OD7B_1 = dfc.List_X(OD7B,4)

OD8B_1 = dfc.List_X(OD8B,4)


df = pandas.read_excel('DATA INPUT ECOLI STAGE
I.xlsx',sheet_name=2,index_col=None)

OD1C = df['Clone 1'].tolist()

OD2C = df['Clone 2'].tolist()

OD3C = df['Clone 3'].tolist()

OD4C = df['Clone 4'].tolist()

OD5C = df['Clone 5'].tolist()

OD6C = df['Clone 6'].tolist()

OD7C = df['Clone 7'].tolist()

OD8C = df['Clone 8'].tolist()


OD1C_1 = dfc.List_X(OD1C,10)

OD2C_1 = dfc.List_X(OD2C,10)

OD3C_1 = dfc.List_X(OD3C,10)
```

```
OD4C_1 = dfc.List_X(OD4C,10)

OD5C_1 = dfc.List_X(OD5C,10)

OD6C_1 = dfc.List_X(OD6C,10)

OD7C_1 = dfc.List_X(OD7C,10)

OD8C_1 = dfc.List_X(OD8C,10)



OD1 = dfc.OD_Sort8(OD1A,OD1B_1,OD1B,OD1C_1,OD1C)

OD2 = dfc.OD_Sort8(OD2A,OD2B_1,OD2B,OD2C_1,OD2C)

OD3 = dfc.OD_Sort8(OD3A,OD3B_1,OD3B,OD3C_1,OD3C)

OD4 = dfc.OD_Sort8(OD4A,OD4B_1,OD4B,OD4C_1,OD4C)

OD5 = dfc.OD_Sort8(OD5A,OD5B_1,OD5B,OD5C_1,OD5C)

OD6 = dfc.OD_Sort8(OD6A,OD6B_1,OD6B,OD6C_1,OD6C)

OD7 = dfc.OD_Sort8(OD7A,OD7B_1,OD7B,OD7C_1,OD7C)

OD8 = dfc.OD_Sort8(OD8A,OD8B_1,OD8B,OD8C_1,OD8C)



GraphOD = input("Would you like to see the OD Graph?")

if GraphOD == 'yes' or GraphOD == 'Yes':

    print("Great")

    x = np.arange(10)

    pp = plt.plot(ODX,OD1,color="red",label="Clone 1")
```

```
    pp = plt.plot(ODX,OD2,color="orange",label="Clone 2")
    pp = plt.plot(ODX,OD3,color="yellow",label="Clone 3")
    pp = plt.plot(ODX,OD4,color="green",label="Clone 4")
    pp = plt.plot(ODX,OD5,color="cyan",label="Clone 5")
    pp = plt.plot(ODX,OD6,color="blue",label="Clone 6")
    pp = plt.plot(ODX,OD7,color="purple",label="Clone 7")
    pp = plt.plot(ODX,OD8,color="pink",label="Clone 8")
    plt.title('Ecoli Stage I OD')
    plt.ylabel('OD')
    plt.xlabel('Time (Hour)')
    plt.legend()
    plt.show()

    Save = input("Would you like to save the OD Graph?")
    if Save == "Yes" or Save == "yes":
        plt.savefig("Ecoli Stage I OD Graph.png")
    else: print ("Sorry for asking...")

else:
    print ("Sorry for asking...")
```

```
df = pandas.read_excel('DATA INPUT ECOLI STAGE
I.xlsx',sheet_name=3,index_col=None)

FBX = df['Time'].tolist()

FB1 = df['Clone 1'].tolist()

FB2 = df['Clone 2'].tolist()

FB3 = df['Clone 3'].tolist()

FB4 = df['Clone 4'].tolist()

FB5 = df['Clone 5'].tolist()

FB6 = df['Clone 6'].tolist()

FB7 = df['Clone 7'].tolist()

FB8 = df['Clone 8'].tolist()



#Area under the curve into fab' concerntration

Default = input('Would you like to proceed with default? (Yes/No)')

if Default == "yes" or Default == "Yes":

    print("Great")

    FAB1 = [(i * 0.00003 + 0.0144) * (8/5) for i in FB1]

    FAB2 = [(i * 0.00003 + 0.0144) * (8/5) for i in FB2]

    FAB3 = [(i * 0.00003 + 0.0144) * (8/5) for i in FB3]
```

```
    FAB4 = [(i * 0.00003 + 0.0144) * (8/5) for i in FB4]

    FAB5 = [(i * 0.00003 + 0.0144) * (8/5) for i in FB5]

    FAB6 = [(i * 0.00003 + 0.0144) * (8/5) for i in FB6]

    FAB7 = [(i * 0.00003 + 0.0144) * (8/5) for i in FB7]

    FAB8 = [(i * 0.00003 + 0.0144) * (8/5) for i in FB8]
if Default == "no" or Default == "No":
    print("Please enter values for A and B for standard y=Ax+B")
    A = input("Value for A")
    B = input("Value for B")
    print("Thank You")
    FAB1 = [(i * int(A) + int(B)) * (8/5) for i in FB1]

    FAB2 = [(i * int(A) + int(B)) * (8/5) for i in FB2]

    FAB3 = [(i * int(A) + int(B)) * (8/5) for i in FB3]

    FAB4 = [(i * int(A) + int(B)) * (8/5) for i in FB4]

    FAB5 = [(i * int(A) + int(B)) * (8/5) for i in FB5]

    FAB6 = [(i * int(A) + int(B)) * (8/5) for i in FB6]

    FAB7 = [(i * int(A) + int(B)) * (8/5) for i in FB7]

    FAB8 = [(i * int(A) + int(B)) * (8/5) for i in FB8]
else:
    print ("Answer not recognised, proceeding with default.")
```

```
    FAB1 = [(i * 0.00003 + 0.0144) * (8/5) for i in FB1]

    FAB2 = [(i * 0.00003 + 0.0144) * (8/5) for i in FB2]

    FAB3 = [(i * 0.00003 + 0.0144) * (8/5) for i in FB3]

    FAB4 = [(i * 0.00003 + 0.0144) * (8/5) for i in FB4]

    FAB5 = [(i * 0.00003 + 0.0144) * (8/5) for i in FB5]

    FAB6 = [(i * 0.00003 + 0.0144) * (8/5) for i in FB6]

    FAB7 = [(i * 0.00003 + 0.0144) * (8/5) for i in FB7]

    FAB8 = [(i * 0.00003 + 0.0144) * (8/5) for i in FB8]



#Show Fab' Graph?

GraphFB = input('Would you like to see the Fab Graph?')

if GraphFB == 'yes' or GraphFB == 'Yes':

    print("Great")

    x = np.arange(10)

    pp = plt.plot(FBX,FAB1,color="red",label="Clone 1")

    pp = plt.plot(FBX,FAB2,color="orange",label="Clone 2")

    pp = plt.plot(FBX,FAB3,color="yellow",label="Clone 3")

    pp = plt.plot(FBX,FAB4,color="green",label="Clone 4")

    pp = plt.plot(FBX,FAB5,color="cyan",label="Clone 5")

    pp = plt.plot(FBX,FAB6,color="blue",label="Clone 6")
```

```
    pp = plt.plot(FBX,FAB7,color="purple",label="Clone 7")
    pp = plt.plot(FBX,FAB8,color="pink",label="Clone 8")

    plt.title('Ecoli Stage I Total Fab Produced per Well')
    plt.ylabel('Total Fab Weight (ug)')
    plt.xlabel('Time (Hour)')
    plt.legend()
    plt.show()

    Save = input("Would you like to save the Fab Graph?")
    if Save == "Yes" or Save == "yes":
        plt.savefig("Ecoli Stage I Fab Graph.png")
else:
    print ("Sorry for asking...")


#Max Values in Fab Lists

FBM1 = dfc.List_Conseq_Sort_3X12(FAB1)
FBM2 = dfc.List_Conseq_Sort_3X12(FAB2)
```

```
FBM3 = dfc.List_Conseq_Sort_3X12(FAB3)

FBM4 = dfc.List_Conseq_Sort_3X12(FAB4)

FBM5 = dfc.List_Conseq_Sort_3X12(FAB5)

FBM6 = dfc.List_Conseq_Sort_3X12(FAB6)

FBM7 = dfc.List_Conseq_Sort_3X12(FAB7)

FBM8 = dfc.List_Conseq_Sort_3X12(FAB8)



FBM1.sort()

FBM2.sort()

FBM3.sort()

FBM4.sort()

FBM5.sort()

FBM6.sort()

FBM7.sort()

FBM8.sort()



FABM1 = FBM1[-1]

FABM2 = FBM2[-1]

FABM3 = FBM3[-1]

FABM4 = FBM4[-1]
```

```
FABM5 = FBM5[-1]

FABM6 = FBM6[-1]

FABM7 = FBM7[-1]

FABM8 = FBM8[-1]


ecstrainsort = [
    (FABM1, 'Clone 1'),
    (FABM2, 'Clone 2'),
    (FABM3, 'Clone 3'),
    (FABM4, 'Clone 4'),
    (FABM5, 'Clone 5'),
    (FABM6, 'Clone 6'),
    (FABM7, 'Clone 7'),
    (FABM8, 'Clone 8'),
]
ecstrainsort.sort()


ecTOP3 = (ecstrainsort[-1],ecstrainsort[-2], ecstrainsort[-3])
print("The top three clones are:")
print("1st:",ecTOP3[0][1])
```

```
print("2nd:",ecTOP3[1][1])

print("3rd:",ecTOP3[2][1])
```

Appendix V: Code for Removing Unwanted *P. pastoris* Clones from Evaluation

```
def Rank_StageI_Remove(a,b,c,d,e,f,g,h):

    a_3 = np.mean([a[0], a[1], a[2]])

    a_6 = np.mean([a[3], a[4], a[5],a[6], a[7], a[8]])

    b_3 = np.mean([b[0], b[1], b[2]])

    b_6 = np.mean([b[3], b[4], b[5],b[6], b[7], b[8]])

    c_3 = np.mean([c[0], c[1], c[2]])

    c_6 = np.mean([c[3], c[4], c[5],c[6], c[7], c[8]])

    d_3 = np.mean([d[0], d[1], d[2]])

    d_6 = np.mean([d[3], d[4], d[5],d[6], d[7], d[8]])

    e_3 = np.mean([e[0], e[1], e[2]])

    e_6 = np.mean([e[3], e[4], e[5],e[6], e[7], e[8]])

    f_3 = np.mean([f[0], f[1], f[2]])

    f_6 = np.mean([f[3], f[4], f[5],f[6], f[7], f[8]])

    g_3 = np.mean([g[0], g[1], g[2]])

    g_6 = np.mean([g[3], g[4], g[5],g[6], g[7], g[8]])

    h_3 = np.mean([h[0], h[1], h[2]])

    h_6 = np.mean([h[3], h[4], h[5],h[6], h[7], h[8]])



    all_values = [(a,"Clone 1"),
```

```
            (b,"Clone 2"),
            (c,"Clone 3"),
            (d,"Clone 4"),
            (e,"Clone 5"),
            (f,"Clone 6"),
            (g,"Clone 7"),
            (h,"Clone 8")]

if h_3 > h_6:
    del all_values[7]
if g_3 > g_6:
    del all_values[6]
if f_3 > f_6:
    del all_values[5]
if e_3 > e_6:
    del all_values[4]
if d_3 > d_6:
    del all_values[3]
if c_3 > c_6:
    del all_values[2]
```

```
if b_3 > b_6:
    del all_values[1]
if a_3 > a_6:
    del all_values[0]


return all_values
```

Appendix VI: Code for Applying Weighting to Highest Fab' Yield and Variance

```
Weight = input('Would you like to proceed with default weighting? (Yes/No)')

if Weight == "no" or Weight == "No":

    print("Please enter weighting for productivity (%)")

    Weighting = input("Productivity Weighting?")

    print("Thank You")

    W1 = int(Weighting)/100

    W2 = 1-W1

    Val1 = W1*ValM1 + W2*Err1

    Val2 = W1*ValM2 + W2*Err2

    Val3 = W1*ValM3 + W2*Err3

else:

    print ("Proceeding with default.")

    Val1 = 0.75*ValM1 + 0.25*Err1

    Val2 = 0.75*ValM2 + 0.25*Err2

    Val3 = 0.75*ValM3 + 0.25*Err3



ecstrainsort = [

      (Val1, CloneA),

      (Val2, CloneB),
```

```
    (Val3, CloneC)]

ecstrainsort.sort()



ecTOP = (ecstrainsort[-1],ecstrainsort[-2], ecstrainsort[-3])

print("The top performing strain is:",ecTOP[0][1])
```

Appendix VII: Database of Defined Functions for MIAMI

```
def Derivative(x,y):
    dx = np.diff(x)
    dy = np.diff(y)
    return dx/dy


def List_AV_3(x,y,z):
    AV1 = np.average([x[0],y[0],z[0]])
    AV2 = np.average([x[1],y[1],z[1]])
    AV3 = np.average([x[2],y[2],z[2]])
    AV4 = np.average([x[3],y[3],z[3]])
    AV5 = np.average([x[4],y[4],z[4]])
    AV6 = np.average([x[5],y[5],z[5]])
    AV7 = np.average([x[6],y[6],z[6]])
    AV8 = np.average([x[7],y[7],z[7]])
    return [AV1,AV2,AV3,AV4,AV5,AV6,AV7,AV8]


def List_AV_3X12(x,y,z):
    AV1 = np.average([x[0],y[0],z[0]])
    AV2 = np.average([x[1],y[1],z[1]])
```

```
    AV3 = np.average([x[2],y[2],z[2]])
    AV4 = np.average([x[3],y[3],z[3]])
    AV5 = np.average([x[4],y[4],z[4]])
    AV6 = np.average([x[5],y[5],z[5]])
    AV7 = np.average([x[6],y[6],z[6]])
    AV8 = np.average([x[7],y[7],z[7]])
    AV9 = np.average([x[8],y[8],z[8]])
    AV10 = np.average([x[9],y[9],z[9]])
    AV11 = np.average([x[10],y[10],z[10]])
    AV12 = np.average([x[11],y[11],z[11]])
    return [AV1,AV2,AV3,AV4,AV5,AV6,AV7,AV8,AV9,AV10,AV11,AV12]


def List_AV_4X6(w,x,y,z):
    AV1 = np.average([w[0],x[0],y[0],z[0]])
    AV2 = np.average([w[1],x[1],y[1],z[1]])
    AV3 = np.average([w[2],x[2],y[2],z[2]])
    AV4 = np.average([w[3],x[3],y[3],z[3]])
    AV5 = np.average([w[4],x[4],y[4],z[4]])
    AV6 = np.average([w[5],x[5],y[5],z[5]])
    return [AV1,AV2,AV3,AV4,AV5,AV6]
```

```
def List_AV_4X8(w,x,y,z):
    AV1 = np.average([w[0],x[0],y[0],z[0]])
    AV2 = np.average([w[1],x[1],y[1],z[1]])
    AV3 = np.average([w[2],x[2],y[2],z[2]])
    AV4 = np.average([w[3],x[3],y[3],z[3]])
    AV5 = np.average([w[4],x[4],y[4],z[4]])
    AV6 = np.average([w[5],x[5],y[5],z[5]])
    AV7 = np.average([w[6],x[6],y[6],z[6]])
    AV8 = np.average([w[7],x[7],y[7],z[7]])
    return [AV1,AV2,AV3,AV4,AV5,AV6,AV7,AV8]


def List_AVER_4X8(w,x,y,z):
    AV1 = np.std([w[0],x[0],y[0],z[0]])
    AV2 = np.std([w[1],x[1],y[1],z[1]])
    AV3 = np.std([w[2],x[2],y[2],z[2]])
    AV4 = np.std([w[3],x[3],y[3],z[3]])
    AV5 = np.std([w[4],x[4],y[4],z[4]])
    AV6 = np.std([w[5],x[5],y[5],z[5]])
    AV7 = np.std([w[6],x[6],y[6],z[6]])
```

```
    AV8 = np.std([w[7],x[7],y[7],z[7]])

    return [AV1,AV2,AV3,AV4,AV5,AV6,AV7,AV8]


def List_Conseq_Sort_3X8(x):

    L123 = np.average([x[0],x[1],x[2]])

    L234 = np.average([x[1],x[2],x[3]])

    L345 = np.average([x[2],x[3],x[4]])

    L456 = np.average([x[3],x[4],x[5]])

    L567 = np.average([x[4],x[5],x[6]])

    L678 = np.average([x[5],x[6],x[7]])

    return [L123,L234,L345,L456,L567,L678]


def List_Conseq_Sort_3X12(x):

    L123 = np.average([x[0],x[1],x[2]])

    L234 = np.average([x[1],x[2],x[3]])

    L345 = np.average([x[2],x[3],x[4]])

    L456 = np.average([x[3],x[4],x[5]])

    L567 = np.average([x[4],x[5],x[6]])

    L678 = np.average([x[5],x[6],x[7]])

    L789 = np.average([x[6],x[7],x[8]])
```

```
    L8910 = np.average([x[7],x[8],x[9]])

    L91011 = np.average([x[8],x[9],x[10]])

    L101112 = np.average([x[9],x[10],x[11]])

    return [L123,L234,L345,L456,L567,L678,L789,L8910,L91011,L101112]


def List_Conseq_Sort_Err_3X8(x):

    L123 = np.std([x[0],x[1],x[2]])

    L234 = np.std([x[1],x[2],x[3]])

    L345 = np.std([x[2],x[3],x[4]])

    L456 = np.std([x[3],x[4],x[5]])

    L567 = np.std([x[4],x[5],x[6]])

    L678 = np.std([x[5],x[6],x[7]])

    return [L123,L234,L345,L456,L567,L678]


def List_Conseq_Sort_WithErr_3X6(x):

    L123 = np.average([x[0],x[1],x[2]])

    L234 = np.average([x[1],x[2],x[3]])

    L345 = np.average([x[2],x[3],x[4]])

    L456 = np.average([x[3],x[4],x[5]])
```

```
    E123 = np.std([x[0],x[1],x[2]])

    E234 = np.std([x[1],x[2],x[3]])

    E345 = np.std([x[2],x[3],x[4]])

    E456 = np.std([x[3],x[4],x[5]])


    all_values = [

        (L123,E123),

        (L234,E234),

        (L345,E345),

        (L456,E456)]


    return all_values


def List_Conseq_Sort_WithErr_3X8(x):

    L123 = np.average([x[0],x[1],x[2]])

    L234 = np.average([x[1],x[2],x[3]])

    L345 = np.average([x[2],x[3],x[4]])

    L456 = np.average([x[3],x[4],x[5]])

    L567 = np.average([x[4],x[5],x[6]])

    L678 = np.average([x[5],x[6],x[7]])
```

```
    E123 = np.std([x[0],x[1],x[2]])

    E234 = np.std([x[1],x[2],x[3]])

    E345 = np.std([x[2],x[3],x[4]])

    E456 = np.std([x[3],x[4],x[5]])

    E567 = np.std([x[4],x[5],x[6]])

    E678 = np.std([x[5],x[6],x[7]])


    all_values = [

            (L123,E123),

            (L234,E234),

            (L345,E345),

            (L456,E456),

            (L567,E567),

            (L678,E678)]


    return all_values


def List_Divide_6(x,y):

    Result = [x[0]/y[0],x[1]/y[1],x[2]/y[2],x[3]/y[3],x[4]/y[4],x[5]/y[5]]
```

```
    return Result


def List_Divide_8(x,y):
    Result = [x[0]/y[0],x[1]/y[1],x[2]/y[2],x[3]/y[3],x[4]/y[4],x[5]/y[5],x[6]/y[6],x[7]/y[7]]
    return Result


def List_Divide_12(x,y):
    Result =
[x[0]/y[0],x[1]/y[1],x[2]/y[2],x[3]/y[3],x[4]/y[4],x[5]/y[5],x[6]/y[6],x[7]/y[7],x[8]/y[8],x[9]/y
[9],x[10]/y[10],x[11]/y[11]]
    return Result


def List_STD_3X8(x,y,z):
    STD1 = np.std([x[0],y[0],z[0]])
    STD2 = np.std([x[1],y[1],z[1]])
    STD3 = np.std([x[2],y[2],z[2]])
    STD4 = np.std([x[3],y[3],z[3]])
    STD5 = np.std([x[4],y[4],z[4]])
    STD6 = np.std([x[5],y[5],z[5]])
    STD7 = np.std([x[6],y[6],z[6]])
```

```
    STD8 = np.std([x[7],y[7],z[7]])
    return [STD1,STD2,STD3,STD4,STD5,STD6,STD7,STD8]

def List_STD_3X12(x,y,z):
    STD1 = np.std([x[0],y[0],z[0]])
    STD2 = np.std([x[1],y[1],z[1]])
    STD3 = np.std([x[2],y[2],z[2]])
    STD4 = np.std([x[3],y[3],z[3]])
    STD5 = np.std([x[4],y[4],z[4]])
    STD6 = np.std([x[5],y[5],z[5]])
    STD7 = np.std([x[6],y[6],z[6]])
    STD8 = np.std([x[7],y[7],z[7]])
    STD9 = np.std([x[8],y[8],z[8]])
    STD10 = np.std([x[9],y[9],z[9]])
    STD11 = np.std([x[10],y[10],z[10]])
    STD12 = np.std([x[11],y[11],z[11]])
    return [STD1,STD2,STD3,STD4,STD5,STD6,STD7,STD8,STD9,STD10,STD11,STD12]

def List_STD_4X8(w,x,y,z):
    STD1 = np.std([w[0],x[0],y[0],z[0]])
```

```
    STD2 = np.std([w[1],x[1],y[1],z[1]])
    STD3 = np.std([w[2],x[2],y[2],z[2]])
    STD4 = np.std([w[3],x[3],y[3],z[3]])
    STD5 = np.std([w[4],x[4],y[4],z[4]])
    STD6 = np.std([w[5],x[5],y[5],z[5]])
    STD7 = np.std([w[6],x[6],y[6],z[6]])
    STD8 = np.std([w[7],x[7],y[7],z[7]])
    return [STD1,STD2,STD3,STD4,STD5,STD6,STD7,STD8]


def List_X(x,V):
    newx = [i * V for i in x]
    return newx


def OD_Sort6(x,y,y_1,z,z_1):
    if (z_1[0] >= 1 or
        z_1[1] >= 1 or
        z_1[2] >= 1 or
        z_1[3] >= 1 or
        z_1[4] >= 1 or
        z_1[5] >= 1):
```

```
    print(z_1, "For OD values larger than 1, please perform further dilution")
    return exit

if y_1[0] > 1:
    del y[0]
    y.insert(0,z[0])
if y_1[1] > 1:
    del y[1]
    y.insert(1,z[1])
if y_1[2] > 1:
    del y[2]
    y.insert(2,z[2])
if y_1[3] > 1:
    del y[3]
    y.insert(3,z[3])
if y_1[4] > 1:
    del y[4]
    y.insert(4,z[4])
if y_1[5] > 1:
    del y[5]
```

```
    y.insert(5,z[5])


if x[0] > 1:
    del x[0]
    x.insert(0,y[0])
if x[1] > 1:
    del x[1]
    x.insert(1,y[1])
if x[2] > 1:
    del x[2]
    x.insert(2,y[2])
if x[3] > 1:
    del x[3]
    x.insert(3,y[3])
if x[4] > 1:
    del x[4]
    x.insert(4,y[4])
if x[5] > 1:
    del x[5]
    x.insert(5,y[5])
```

```
    return x


def OD_Sort8(x,y,y_1,z,z_1):
    if (z_1[0] >= 1 or
        z_1[1] >= 1 or
        z_1[2] >= 1 or
        z_1[3] >= 1 or
        z_1[4] >= 1 or
        z_1[5] >= 1 or
        z_1[6] >= 1 or
        z_1[7] >= 1):
        print(z_1, "For OD values larger than 1, please perform further dilution")
        return exit


    if y_1[0] > 1:
        del y[0]
        y.insert(0,z[0])
    if y_1[1] > 1:
        del y[1]
```

```
    y.insert(1,z[1])
if y_1[2] > 1:
    del y[2]
    y.insert(2,z[2])
if y_1[3] > 1:
    del y[3]
    y.insert(3,z[3])
if y_1[4] > 1:
    del y[4]
    y.insert(4,z[4])
if y_1[5] > 1:
    del y[5]
    y.insert(5,z[5])
if y_1[6] > 1:
    del y[6]
    y.insert(6,z[6])
if y_1[7] > 1:
    del y[7]
    y.insert(7,z[7])
```

```
if x[0] > 1:
    del x[0]
    x.insert(0,y[0])
if x[1] > 1:
    del x[1]
    x.insert(1,y[1])
if x[2] > 1:
    del x[2]
    x.insert(2,y[2])
if x[3] > 1:
    del x[3]
    x.insert(3,y[3])
if x[4] > 1:
    del x[4]
    x.insert(4,y[4])
if x[5] > 1:
    del x[5]
    x.insert(5,y[5])
if x[6] > 1:
    del x[6]
```

```
        x.insert(6,y[6])
    if x[7] > 1:
        del x[7]
        x.insert(7,y[7])

    return x

def OD_Sort12(x,y,y_1,z,z_1):
    if (z_1[0] >= 1 or
        z_1[1] >= 1 or
        z_1[2] >= 1 or
        z_1[3] >= 1 or
        z_1[4] >= 1 or
        z_1[5] >= 1 or
        z_1[6] >= 1 or
        z_1[7] >= 1 or
        z_1[8] >= 1 or
        z_1[9] >= 1 or
        z_1[10] >= 1 or
        z_1[11] >= 1):
```

```
    print(z_1, "For OD values larger than 1, please perform further dilution")
    return exit


if y_1[0] > 1:
    del y[0]
    y.insert(0,z[0])
if y_1[1] > 1:
    del y[1]
    y.insert(1,z[1])
if y_1[2] > 1:
    del y[2]
    y.insert(2,z[2])
if y_1[3] > 1:
    del y[3]
    y.insert(3,z[3])
if y_1[4] > 1:
    del y[4]
    y.insert(4,z[4])
if y_1[5] > 1:
    del y[5]
```

```
    y.insert(5,z[5])
if y_1[6] > 1:
    del y[6]
    y.insert(6,z[6])
if y_1[7] > 1:
    del y[7]
    y.insert(7,z[7])
if y_1[8] > 1:
    del y[8]
    y.insert(8,z[8])
if y_1[9] > 1:
    del y[9]
    y.insert(9,z[9])
if y_1[10] > 1:
    del y[10]
    y.insert(10,z[10])
if y_1[11] > 1:
    del y[11]
    y.insert(11,z[11])
```

```
if x[0] > 1:
    del x[0]
    x.insert(0,y[0])
if x[1] > 1:
    del x[1]
    x.insert(1,y[1])
if x[2] > 1:
    del x[2]
    x.insert(2,y[2])
if x[3] > 1:
    del x[3]
    x.insert(3,y[3])
if x[4] > 1:
    del x[4]
    x.insert(4,y[4])
if x[5] > 1:
    del x[5]
    x.insert(5,y[5])
if x[6] > 1:
    del x[6]
```

```
        x.insert(6,y[6])
    if x[7] > 1:
        del x[7]
        x.insert(7,y[7])
    if x[8] > 1:
        del x[8]
        x.insert(8,y[8])
    if x[9] > 1:
        del x[9]
        x.insert(9,y[9])
    if x[10] > 1:
        del x[10]
        x.insert(10,y[10])
    if x[11] > 1:
        del x[11]
        x.insert(11,y[11])

    return x

def Rank_StageI_Remove(a,b,c,d,e,f,g,h):
```

```
a_3 = np.mean([a[0], a[1], a[2]])

a_6 = np.mean([a[3], a[4], a[5],a[6], a[7], a[8]])

b_3 = np.mean([b[0], b[1], b[2]])

b_6 = np.mean([b[3], b[4], b[5],b[6], b[7], b[8]])

c_3 = np.mean([c[0], c[1], c[2]])

c_6 = np.mean([c[3], c[4], c[5],c[6], c[7], c[8]])

d_3 = np.mean([d[0], d[1], d[2]])

d_6 = np.mean([d[3], d[4], d[5],d[6], d[7], d[8]])

e_3 = np.mean([e[0], e[1], e[2]])

e_6 = np.mean([e[3], e[4], e[5],e[6], e[7], e[8]])

f_3 = np.mean([f[0], f[1], f[2]])

f_6 = np.mean([f[3], f[4], f[5],f[6], f[7], f[8]])

g_3 = np.mean([g[0], g[1], g[2]])

g_6 = np.mean([g[3], g[4], g[5],g[6], g[7], g[8]])

h_3 = np.mean([h[0], h[1], h[2]])

h_6 = np.mean([h[3], h[4], h[5],h[6], h[7], h[8]])



all_values = [(a,"Clone 1"),

        (b,"Clone 2"),
```

```
            (c,"Clone 3"),
            (d,"Clone 4"),
            (e,"Clone 5"),
            (f,"Clone 6"),
            (g,"Clone 7"),
            (h,"Clone 8")]

if h_3 > h_6:
    del all_values[7]
if g_3 > g_6:
    del all_values[6]
if f_3 > f_6:
    del all_values[5]
if e_3 > e_6:
    del all_values[4]
if d_3 > d_6:
    del all_values[3]
if c_3 > c_6:
    del all_values[2]
if b_3 > b_6:
```

```
        del all_values[1]
    if a_3 > a_6:
        del all_values[0]
    return all_values


def Least_Square_Fit_A(x,y,up_lim,low_lim):
    i= low_lim - 1
    n= up_lim -1

    sum_xxy = x[int(i)] * x[int(i)] * y[int(i)]
    sum_ylny = y[int(i)] * np.log(y[int(i)])
    sum_xy = x[int(i)]* y[int(i)]
    sum_xylny = x[int(i)]* y[int(i)] * np.log(y[int(i)])
    sum_y = y[int(i)]


    i_1 = int(i) + 1

    while n >= i_1:
        sum_xxy = sum_xxy + x[int(i_1)] * x[int(i_1)] * y[int(i_1)]
```

```
        sum_ylny = sum_ylny + y[int(i_1)] * np.log(y[int(i_1)])
        sum_xy = sum_xy + x[int(i_1)]* y[int(i_1)]
        sum_xylny = sum_xylny + x[int(i_1)]* y[int(i_1)] * np.log(y[int(i_1)])
        sum_y = sum_y + y[int(i_1)]
        i_1 = i_1 + 1

    a = (sum_xxy*sum_ylny - sum_xy*sum_xylny) / (sum_y * sum_xxy - sum_xy * sum_xy)
    true_A = np.exp(a)
    return true_A

def Least_Square_Fit_B(x,y,up_lim,low_lim):
    i= low_lim - 1
    n= up_lim -1

    sum_xxy = x[int(i)] * x[int(i)] * y[int(i)]
    sum_ylny = y[int(i)] * np.log(y[int(i)])
    sum_xy = x[int(i)]* y[int(i)]
    sum_xylny = x[int(i)]* y[int(i)] * np.log(y[int(i)])
    sum_y = y[int(i)]
    i_1 = int(i) + 1
```

```
while n >= i_1:
    sum_xxy = sum_xxy + x[int(i_1)] * x[int(i_1)] * y[int(i_1)]
    sum_ylny = sum_ylny + y[int(i_1)] * np.log(y[int(i_1)])
    sum_xy = sum_xy + x[int(i_1)]* y[int(i_1)]
    sum_xylny = sum_xylny + x[int(i_1)]* y[int(i_1)] * np.log(y[int(i_1)])
    sum_y = sum_y + y[int(i_1)]
    i_1 = i_1 + 1
b = (sum_y*sum_xylny - sum_xy*sum_ylny) / (sum_y*sum_xxy - sum_xy*sum_xy)
return b
```