THE UNIVERSITY OF EDINBURGH

Balancing Complexity and Efficiency in Microbial Factories with Machine Learning

Nicholas Goguen-Compagnoni

4th Year Project Report
Biotechnology
School of Biological Sciences
University of Edinburgh

2024

Abstract

Scientists can synthesize products from bacterial cells by hijacking their metabolism to create microbial factories. Engineering a metabolic pathway with optimal parameters is expensive when the design space is very large. Recent works have used computational models to represent metabolic pathways and machine learning algorithms to search the space, but the models do not fully represent cellular growth constraints and the search space remains computationally expensive to explore. This work optimizes the p-Aminostyrene (p-AS) pathway by adding total proteome and proteome rate constraints to the optimization objective, in addition to a regularization term which accounts for model complexity. All three additions to the model significantly change the distribution of optimal circuit architectures as their values are swept. This work secondarily improves computational efficiency and robustness by implementing the model in Julia, a high-level, compiled scientific programming language.

Table of Contents

List of Abbreviations

aREDs aptazyme-Regulated Expression Devices

BayesOpt Bayesian Optimization

DNA Deoxyribonucleic Acid

FBDF (Fixed-leading coefficient Backward Differentiation Formula)

LAAO L-Amino Acid Oxidase

LSODA Livermore Solver for Ordinary Differential Equations

ODE Ordinary Differential Equation

p-ACA p-Aminocinnamic Acid

p-AF p-Aminophenylalanine

p-AS p-Aminostyrene

QNDF Quasi-constant timestep Numerical Differentiation Formula

RNA Ribonucleic Acid

mRNA Messenger RNA

tRNA Transfer RNA

Chapter 1

Introduction

Metabolic engineering is the deliberate altering of cellular metabolism to produce specific compounds [1]. Metabolic engineering is done with recombinant Deoxyribonucleic Acid (DNA) technology to manipulate cell DNA, turning it into a microbial factory of foreign protein products [2]. Synthetic biology provides a set of abstractions to manage the complexity of microbial systems. This allows metabolic engineers to design function through genetic, Ribonucleic Acid (RNA), or protein circuits, which can be combined to create a circuit architecture [3]. The maximization of product flux in a microbial factory is a complex optimization problem, often working across genetic, transcriptional, translational, and posttranslational control [4, 5], which operate on different time and concentration scales [6].

1.1 p-AS Pathway

The p-AS pathway is notable for its complexity and size, surpassing other pathways studied [7]. The intermediates, enzymes, aptazyme-Regulated Expression Devices (aREDs), and Messenger RNA (mRNA) needed to design a synthetic pathway for p-AS in *Escherichia. coli* are already understood [Figure 1.1].

The aREDs can activate, repress, or have no effect on mRNA expression of key enzymes [8]. A previously engineered genetic control framework for p-AS uses three points of control: the Pap operon enzymes (papA, papB, and papC) that convert chorismate to p-AF, controlling pathway flux; LAAO, an enzyme that converts p-AF to p-ACA, producing a toxic intermediate; and the efflux pump that exports p-ACA from the cell. The combinations for regulating these three nodes make up the discrete decision variables for the dynamic control of the pathway [9]. Beyond the architecture design, each mRNA has a range of continuous decision variables which represent the kinetic parameters of the design space [7]. In total, the p-AS pathway has 16 kinetic parameters and 27 possible architectures. The pathway presents some additional considerations, including the toxicity of p-ACA, LAAO, and the efflux pump, and the leaking of p-AF from cells [8]. Importantly, the processes in the p-AS pathway operate across multiple scales, such as metabolic processes, mRNA dynamics, and protein folding [9]. All of these processes can be described mathematically, through a set of Ordinary Differential Equations (ODEs) as a model of the pathway.

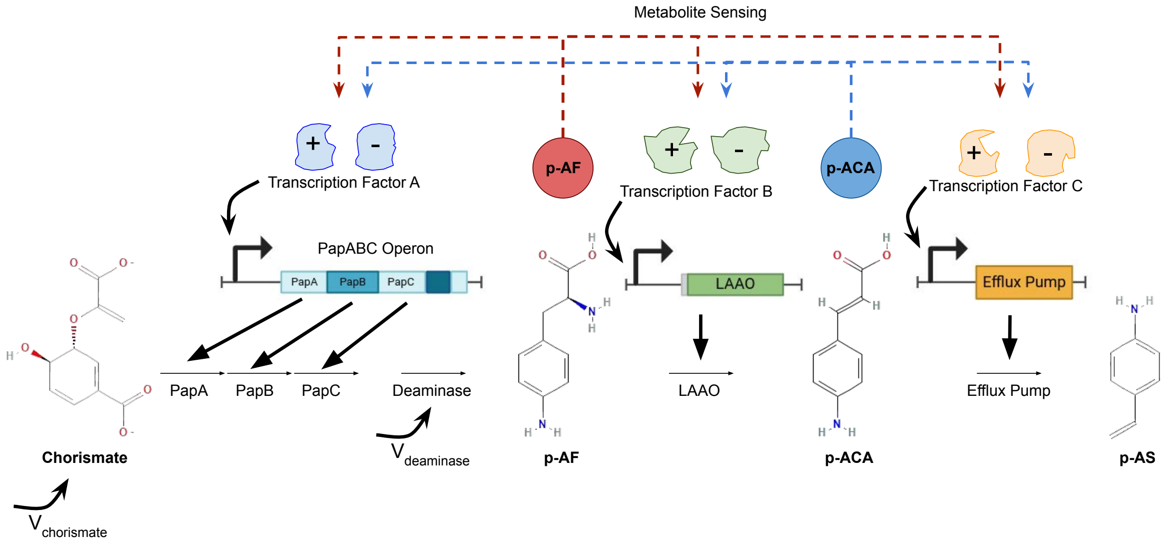

Figure 1.1: **Graphical abstract of the engineered p-AS pathway and the dynamic control system design space.** In this pathway, the substrate Chorismate is converted to p-Aminophenylalanine (p-AF) by the PapABC operon and Deaminase. p-AF is converted to p-Aminocinnamic Acid (p-ACA) by L-Amino Acid Oxidase (LAAO). p-ACA is finally converted to p-AS by the efflux pump. Each set of enzymes can be dynamically controlled through metabolite sensing, where p-AF (red) or p-ACA (blue) are sensed by activating or repressing transcription factors called aREDs, which can create positive or negative feedback loops. The model compresses the dynamic controllers of p-AF and p-ACA, aREDs, and transcription factors into a single sigmoid function. Pathway intermediates are bolded.

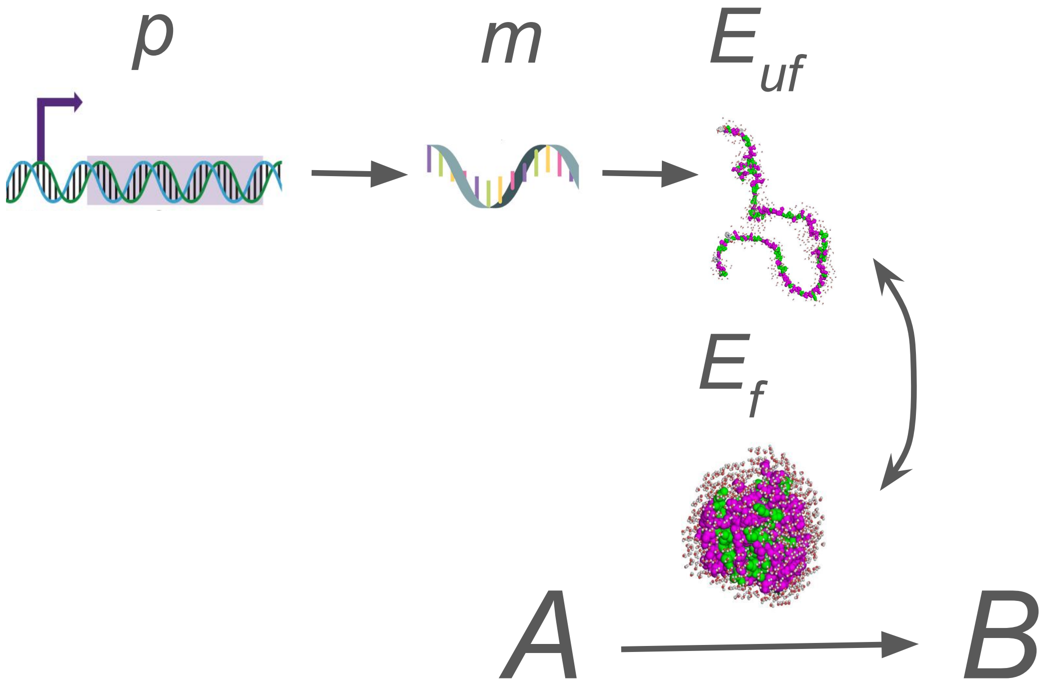

Figure 1.2: **Graphical representation of a simple pathway.** A substrate A is converted to a product B when catalyzed by the folded enzyme E_f, which is folded from the the unfolded enzyme E_{uf} as described by a folding equilibrium equation. E_{uf} is translated from the mRNA m at a rate relative to the concentration of m. m is transcribed relative to the strength of the gene's promoter, p. Together these elements rationalize the equations which define enzyme expression rate, translation rate, and promoter production rate in the p-AS model.

1.2 Ordinary Differential Equations

In metabolic engineering, sets of ODEs can model microbial cells to facilitate predictions to changes in conditions without the high costs of wet-lab experiments [10]. With the genetic control framework of the p-AS pathway in mind, the metabolic dynamics can be computationally modeled using a set of ODEs to represent enzyme, metabolite, and transcriptional element mass balance equations [7]. Given an architecture and a set of input parameters, the flux of each pathway metabolite can be modeled over a time series to find how the mass of each element changes over time [11].

Before we define the ODEs used in the p-AS pathway, we will use an example pathway to introduce the equations simply. [Figure 1.2] demonstrates the production of B from A catalyzed by E_f, which is folded from E_{uf}, which is translated relative to the concentration of m, which is transcribed relative to the strength of promoter p. To find the final concentration of B, we can model these elements as a set of ODEs:

$$
\begin{aligned}
\frac{\mathrm{d}p}{\mathrm{d}t} &= \text{DNA duplication rate} * [p] - \text{dilution rate} * [p], \\
\frac{\mathrm{d}m}{\mathrm{d}t} &= [m] - [m] * \text{dilution rate} - [m] * \text{mRNA degradation rate}, \\
\frac{\mathrm{d}E_{uf}}{\mathrm{d}t} &= \frac{[m]}{\text{translation rate} + \frac{\text{mRNA length}}{\text{ribosome elongation rate}}} - [E_{uf}] * \text{protein folding rate} \\
&\quad - [E_{uf}] * \text{dilution rate} - [E_{uf}] * \text{protein degradation rate}, \\
\frac{\mathrm{d}E_f}{\mathrm{d}t} &= [E_{uf}] * \text{protein folding rate} - [E_f] * \text{dilution rate} - [E_f] * \text{protein degradation rate},
\end{aligned}
\tag{1.1}
$$

where the DNA duplication rate, dilution rate, mRNA degradation rate, translation rate, mRNA length, ribosome elongation rate, protein folding rate, and protein degradation rate are all constants defined in the literature which may vary depending on the host organism. Given give a set of initial conditions, we can see how the concentrations of each element changes over time.

It is important to note that more computational resources are required as more equations are added to the set of ODEs, making large design spaces computationally expensive to optimize [6, 12]. Additionally, a model by definition cannot fully represent all aspects that have an effect on a microbial cell.

1.3 Bayesian Optimization

The design space of the p-AS pathway is much larger than that of the simple pathway. This is unfeasible to search using brute-force methods. The multi-scaled nature of the p-AS pathway makes current optimization methods such as grid-based search and genetic algorithms unusable, as the computational cost of simulating the pathway for thousands or millions of iterations is prohibitively high [7], and each set of ODE solutions become numerically stiff [13]. Previous work has focused on optimizing metabolic pathways using gradient-based optimization algorithms [6, 14], but these algorithms tend to converge to local optima of the search spaces [15].

Bayesian Optimization (BayesOpt) is a machine learning method for finding the set of parameters which returns the minimum of a given function [16]. BayesOpt is a particularly good method for navigating the metabolically engineered pathway of p-AS because the design space has an unknown convexity, is large but traversable, and each computation of the objective function is expensive due to the multiple scales of the pathway and the small step sizes required to accurately solve the ODEs [7]. Because of the nature of kinetic parameters, there is no underlying equation that dictates the structure of the search space, since tiny variations in kinetic parameter values can drastically affect pathway flux.

1.4 Complexity

In the context of synthetic biology, complexity in a genetic circuit architecture is defined as the number of transcription factors and ribosome binding sites required to perform dynamic control [4, 17].

When optimizing product flux of microbial factories *in vitro*, the cost of engineering the genetic circuit increases as additional activator or repressor elements are added [6]. Therefore, for BayesOpt to find the most cost-effective architectures to engineer *in vitro*, there must be a function to penalize increased numbers of loops, biosensors, and controlled promoters [6]. The addition of this regularization term can allow BayesOpt to return architectures that are cost-effective: with the highest overall metabolic production and the lowest cost to engineer. Other works base their complexity values from databases like LASER [18], but because all circuit architecture complexity in our model comes from the number of loops and biosensors, we define our own complexity function.

1.5 Aims and Objectives

This work aims to improve the objective function's comprehensibility by:

- Defining a total proteome constraint that is consistent with the maximum proteome production of *E. coli* strains
- Defining a proteome rate constraint that is consistent with the maximum proteome rate production of *E. coli* strains
- Defining a complexity regularization term to account for the higher engineering costs as circuit complexity increases

For this work, constraints are switches which can take one of two values depending on conditions in the model, while regularization terms are continuous values dependent on inputs to the model.

Secondarily, this work aims to implement the p-AS model and the BayesOpt loop in Julia. It is expected that Julia will improve the computational speed and robustness of ODEs solving compared to previous work on the p-AS, which was implemented in Python [7], which can be confirmed through runtime and robustness benchmarking. This is especially important given the intensive computational resources that ODE solvers and BayesOpt require.

Chapter 2

Methods

ODEs are an accepted and widely used method to represent biological systems computationally [10, 19–21]. In this chapter we define a set of ODEs to apply to the previously discussed p-AS pathway.

2.1 p-AS Model

The p-AS model is based on the work done in Merzbacher et al. [7] and Carothers and Stevens [9]. There are 16 continuous decision variables (each kinetic parameter) and 27 discrete decision variables (each unique circuit architecture) which make up the p-AS search space [Table 2.1]. The p-AS pathway includes two metabolites which can act on transcription factors: p-AF and p-ACA. We simplify the dynamic controllers of p-AF and p-ACA, aREDs, and transcription factors by compressing mRNA activation and repression to p-AF and p-ACA using the equation:

$$u(x, k_{i,j}, \theta_{i,j}) = \begin{cases} k_{i,j} & \text{no control (N),} \\ \frac{k_{i,j}x^2}{\theta_{i,j}^2 + x^2} & \text{activation (A),} \\ \frac{k_{i,j}\theta_{i,j}^2}{\theta_{i,j}^2 + x^2} & \text{repression (R),} \end{cases} \tag{2.1}$$

where x is the concentration of p-AF or p-ACA and $k_{i,j}$ and $\theta_{i,j}$ are the respective continuous parameters pertaining to the chosen architecture, where i refers to the genetic element being targeted (PapABC Operon, PapA, PapB, PapC, LAAO, or Efflux Pump), and j refers to the metabolite sensor (p-AF or p-ACA).

The discrete architecture parameters are defined in three parts: the PapABC operon, the LAAO gene, and the Efflux Pump gene. Architectures which result in a positive feedback loop are excluded, leaving a combination of 27 possible architectures.

- The PapABC operon can have no control, repression by p-AF, or repression by p-ACA,
- The LAAO mRNA construct can have no control, activation by p-AF, or repression by p-ACA,
- The Efflux Pump mRNA construct can have no control, activation by p-AF, or activation by p-ACA [Table 2.1].

We present a notation for these architectures:

$$\{\mathrm{N}|\mathrm{R}_{\text{p-AF}}|\mathrm{R}_{\text{p-ACA}}\},\ \{\mathrm{N}|\mathrm{A}_{\text{p-AF}}|\mathrm{R}_{\text{p-ACA}}\},\ \{\mathrm{N}|\mathrm{A}_{\text{p-AF}}|\mathrm{A}_{\text{p-ACA}}\},$$

where the first element controls the PapABC operon, the second element controls the LAAO gene, and the third element controls the Efflux Pump gene. N, A, and R are the no control, activation, and repression controls, and the p-AF p-ACA subscripts dictate which metabolite is sensing the control mechanism. We show all of the possible architectures and a graphical representation in [Figure 2.1].

Initial Conditions	$x(0) = 0$mM (for all metabolites) $e(0) = 0$mM (for all enzymes)
Decision Variables (Continuous)	$\theta_{\text{PapABC Operon, p-AF}}$, $\theta_{\text{PapABC Operon, p-ACA}}$, $\theta_{\text{LAAO, p-AF}}$, $\theta_{\text{LAAO, p-ACA}}$, $\theta_{\text{Efflux Pump, p-AF}}$, $\theta_{\text{Efflux Pump, p-ACA}}$, $k_{\text{PapA, p-AF}}$, $k_{\text{PapA, p-ACA}}$, $k_{\text{PapB, p-AF}}$, $k_{\text{PapB, p-ACA}}$, $k_{\text{PapC, p-AF}}$, $k_{\text{PapC, p-ACA}}$, $k_{\text{LAAO, p-AF}}$, $k_{\text{LAAO, p-ACA}}$, $k_{\text{Efflux Pump, p-AF}}$, $k_{\text{Efflux Pump, p-ACA}}$
Decision Variables (Discrete)	$[\mathrm{N}|\mathrm{R}_{\text{p-AF}}|\mathrm{R}_{\text{p-ACA}}]$, $[\mathrm{N}|\mathrm{A}_{\text{p-AF}}|\mathrm{R}_{\text{p-ACA}}]$, $[\mathrm{N}|\mathrm{A}_{\text{p-AF}}|\mathrm{A}_{\text{p-ACA}}]$
Pathway Metabolites	Chorismate, PA1, PA2, PA3 p-AF, p-ACA, p-AS
Pathway Enzymes	PapA, PapB, PapC, LAAO Efflux Pump, Deaminase
Integration Time	$1.73 \cdot 10^5$s

Table 2.1: P-Aminostryene model summary.

There are 7 pathway metabolites, 3 DNA promoter elements, 5 mRNA transcripts, 4 unfolded enzymes, 5 folded enzymes, and a folded and unfolded efflux pump protein described in the mass balance equations. The metabolite mass balance equations for the p-AS model are:

$$\begin{aligned}
\frac{\mathrm{dchorismate}}{\mathrm{d}t} &= \mathrm{V}_{\text{chorismate}}\tau - [\text{chorismate}]\cdot mm(\text{papA}) - \delta\cdot[\text{chorismate}],\\
\frac{\mathrm{dpA1}}{\mathrm{d}t} &= [\text{chorismate}]\cdot mm(\text{papA}) - [\text{pA1}]\cdot mm(\text{papB}) - \delta\cdot\text{pA1},\\
\frac{\mathrm{dpA2}}{\mathrm{d}t} &= [\text{pA1}]\cdot mm(\text{papB}) - [\text{pA2}]\cdot mm(\text{papC}) - \delta\cdot\text{pA2},\\
\frac{\mathrm{dpA3}}{\mathrm{d}t} &= [\text{pA2}]\cdot mm(\text{papC}) - f(\text{deaminase, pA3}) - \delta\cdot\text{pA3},\\
\frac{\text{dp-AF}}{\mathrm{d}t} &= f(\text{deaminase, pA3}) - f(\text{LAAO, p-AF}) - \delta\cdot\text{p-AF} - \mathrm{L},\\
\frac{\text{dp-ACA}}{\mathrm{d}t} &= f(\text{LAAO, p-AF}) - f(\mathrm{P}_{\text{efflux}}, \text{p-ACA}) - \delta\cdot\text{p-ACA},\\
\frac{\text{dp-AS}}{\mathrm{d}t} &= f(\mathrm{P}_{\text{efflux}}, \text{p-ACA}).
\end{aligned} \tag{2.2}$$

The constant L is the rate of loss of p-AF through the leaky cellular membrane, and $\mathrm{V}_{\text{chorismate}}$ is a constant parameter defined in [Table 2.2]. The constant τ is the toxicity factor which scales concentra-

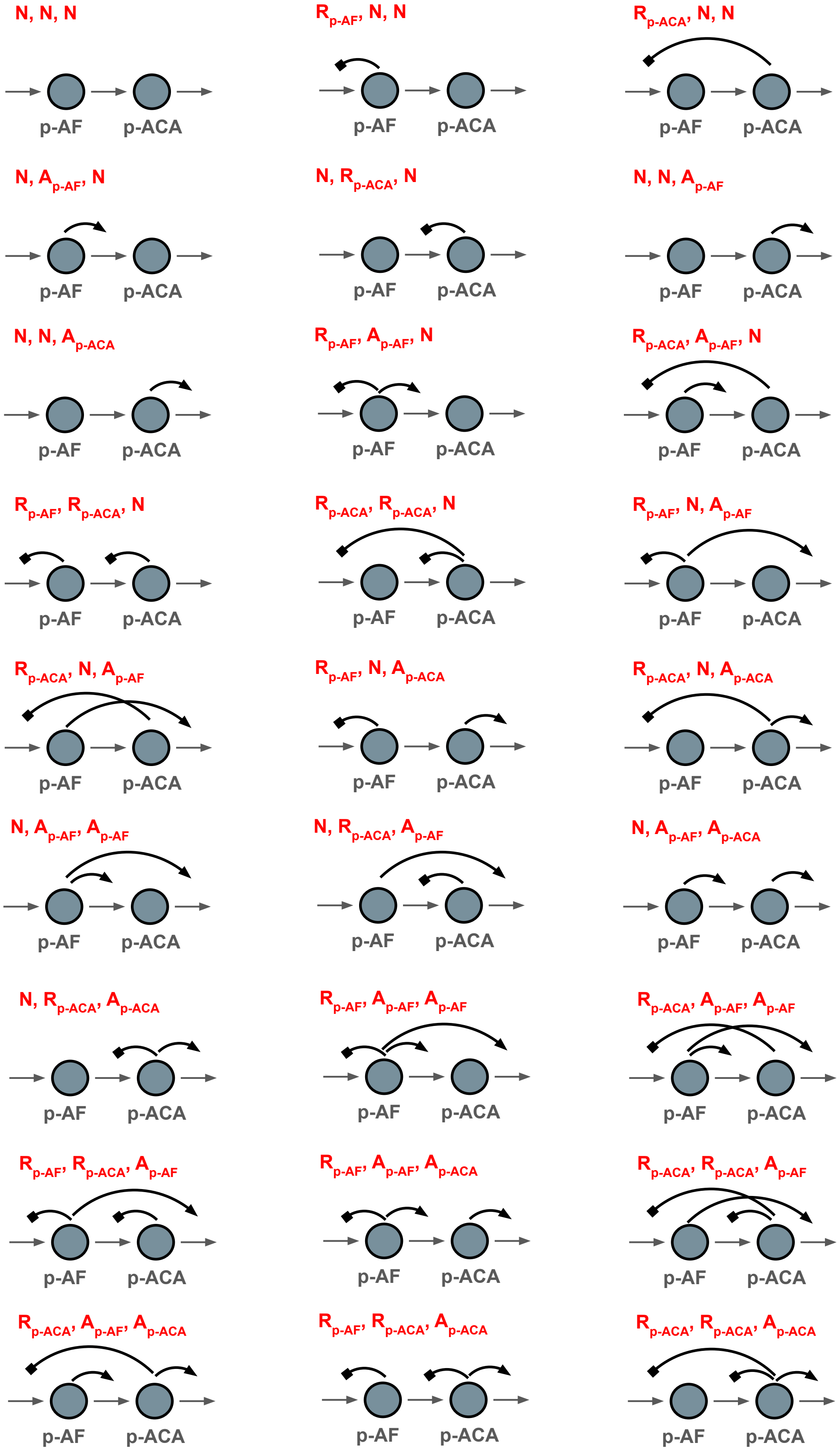
N, N, N
p-AF p-ACA
$R_{p\text{-}AF}$, N, N
p-AF p-ACA
$R_{p\text{-}ACA}$, N, N
p-AF p-ACA
N, $A_{p\text{-}AF}$, N
p-AF p-ACA
N, $R_{p\text{-}ACA}$, N
p-AF p-ACA
N, N, $A_{p\text{-}AF}$
p-AF p-ACA
N, N, $A_{p\text{-}ACA}$
p-AF p-ACA
$R_{p\text{-}AF}$, $A_{p\text{-}AF}$, N
p-AF p-ACA
$R_{p\text{-}ACA}$, $A_{p\text{-}AF}$, N
p-AF p-ACA
$R_{p\text{-}AF}$, $R_{p\text{-}ACA}$, N
p-AF p-ACA
$R_{p\text{-}ACA}$, $R_{p\text{-}ACA}$, N
p-AF p-ACA
$R_{p\text{-}AF}$, N, $A_{p\text{-}AF}$
p-AF p-ACA
$R_{p\text{-}ACA}$, N, $A_{p\text{-}AF}$
p-AF p-ACA
$R_{p\text{-}AF}$, N, $A_{p\text{-}ACA}$
p-AF p-ACA
$R_{p\text{-}ACA}$, N, $A_{p\text{-}ACA}$
p-AF p-ACA
N, $A_{p\text{-}AF}$, $A_{p\text{-}AF}$
p-AF p-ACA
N, $R_{p\text{-}ACA}$, $A_{p\text{-}AF}$
p-AF p-ACA
N, $A_{p\text{-}AF}$, $A_{p\text{-}ACA}$
p-AF p-ACA
N, $R_{p\text{-}ACA}$, $A_{p\text{-}ACA}$
p-AF p-ACA
$R_{p\text{-}AF}$, $A_{p\text{-}AF}$, $A_{p\text{-}AF}$
p-AF p-ACA
$R_{p\text{-}ACA}$, $A_{p\text{-}AF}$, $A_{p\text{-}AF}$
p-AF p-ACA
$R_{p\text{-}AF}$, $R_{p\text{-}ACA}$, $A_{p\text{-}AF}$
p-AF p-ACA
$R_{p\text{-}AF}$, $A_{p\text{-}AF}$, $A_{p\text{-}ACA}$
p-AF p-ACA
$R_{p\text{-}ACA}$, $R_{p\text{-}ACA}$, $A_{p\text{-}AF}$
p-AF p-ACA
$R_{p\text{-}ACA}$, $A_{p\text{-}AF}$, $A_{p\text{-}ACA}$
p-AF p-ACA
$R_{p\text{-}AF}$, $R_{p\text{-}ACA}$, $A_{p\text{-}ACA}$
p-AF p-ACA
$R_{p\text{-}ACA}$, $R_{p\text{-}ACA}$, $A_{p\text{-}ACA}$
p-AF p-ACA

Figure 2.1: **All 27 possible architectures visualized.** The two pathway intermediates (p-AF, p-ACA) are represented as circles, and point to the genetic element they control as represented by the arrows dividing the pathway intermediates, where the first is the PapABC operon, the second is the LAAO gene, and the third is the Efflux Pump. Activation is represented by an arrow, and repression is represented as a square.

tions to account for intermediate metabolite toxicity. The constant δ is the dilution rate due to cellular growth. The functions *mm* and f are the Michaelis-Menten equation and its variation, respectively:

$$\begin{aligned}
&mm(e) = \tfrac{k_{\text{cat}}e}{k_{\text{m}}+e} \\
&\text{where:} \\
&e \in \{\text{papA, papB, papC}\} \\
&f(e,x) = k_{\text{cat}}e\tfrac{\frac{x}{\text{N}_\text{A}\text{Vol}_\text{cell}}}{k_\text{m}+\frac{x}{\text{N}_\text{A}\text{Vol}_\text{cell}}}\tau \\
&\text{where:} \\
&e \in \{\text{deaminase, LAAO, P}_\text{efflux}\} \\
&x \in \{\text{pA3, p-AF, p-ACA}\}
\end{aligned} \tag{2.3}$$

The promoter and mRNA transcript concentrations are modeled separately. Their mass-balance equations are:

$$\begin{aligned}
\frac{\text{dPr}_1}{\text{d}t} &= \text{Pr}_1\beta - \delta\cdot\text{Pr}_1, \\
\frac{\text{dPr}_2}{\text{d}t} &= \text{Pr}_2\beta - \delta\cdot\text{Pr}_2, \\
\frac{\text{dPr}_3}{\text{d}t} &= \text{Pr}_3\beta - \delta\cdot\text{Pr}_3, \\
\frac{\text{dmRNA}_\text{papA}}{\text{d}t} &= \omega_1 u(\text{p-AF}, k_{1,\text{ p-AF}}, \theta_{1,\text{ p-AF}}) + (1-\omega_1)u(\text{p-ACA}, k_{1,\text{ p-ACA}}, \theta_{1,\text{ p-ACA}}) \\
&\quad - \delta\cdot\text{mRNA}_\text{papA} - \tau\cdot\mu\cdot\text{mRNA}_\text{papA}, \\
\frac{\text{dmRNA}_\text{papB}}{\text{d}t} &= \omega_1 u(\text{p-AF}, k_{2,\text{ p-AF}}, \theta_{2,\text{ p-AF}}) + (1-\omega_1)u(\text{p-ACA}, k_{2,\text{ p-ACA}}, \theta_{2,\text{ p-ACA}}) \\
&\quad - \delta\cdot\text{mRNA}_\text{papB} - \tau\cdot\mu\cdot\text{mRNA}_\text{papB}, \\
\frac{\text{dmRNA}_\text{papC}}{\text{d}t} &= \omega_1 u(\text{p-AF}, k_{3,\text{ p-AF}}, \theta_{3,\text{ p-AF}}) + (1-\omega_1)u(\text{p-ACA}, k_{3,p-ACA}, \theta_{3,\text{ p-ACA}}) \\
&\quad - \delta\cdot\text{mRNA}_\text{papC} - \tau\cdot\mu\cdot\text{mRNA}_\text{papC}, \\
\frac{\text{dmRNA}_\text{LAAO}}{\text{d}t} &= \omega_2 u(\text{p-AF}, k_{4,\text{ p-AF}}, \theta_{4,\text{ p-AF}}) + (1-\omega_2)u(\text{p-ACA}, k_{4,\text{ p-ACA}}, \theta_{4,\text{ p-ACA}}) \\
&\quad - \delta\cdot\text{mRNA}_\text{LAAO} - \tau\cdot\mu\cdot\text{mRNA}_\text{LAAO}, \\
\frac{\text{dmRNA}_{\text{P}_\text{efflux}}}{\text{d}t} &= \omega_3 u(\text{p-AF}, k_{5,\text{ p-AF}}, \theta_{5,\text{ p-AF}}) + (1-\omega_3)u(\text{p-ACA}, k_{5,\text{ p-ACA}}, \theta_{5,\text{ p-ACA}}) \\
&\quad - \delta\cdot\text{mRNA}_{\text{P}_\text{efflux}} - \tau\cdot\mu\cdot\text{mRNA}_{\text{P}_\text{efflux}}.
\end{aligned} \tag{2.4}$$

The constant β is the DNA duplication rate. The constant μ is the mRNA degradation rate, which is assumed to be constant for all mRNA. The function $u(x,k,\theta)$ is the control function defined in [Equation 2.1]. ω_i defines which metabolite controls each promoter:

$$\omega_1 = \begin{cases} 0 & \text{if papABC is controlled by p-AF,} \\ 1 & \text{if papABC is controlled by p-ACA,} \end{cases}$$
$$\omega_2 = \begin{cases} 0 & \text{if LAAO is controlled by p-AF,} \\ 1 & \text{if LAAO is controlled by p-ACA,} \end{cases} \tag{2.5}$$
$$\omega_3 = \begin{cases} 0 & \text{if } \mathrm{P_{efflux}} \text{ is controlled by p-AF,} \\ 1 & \text{if } \mathrm{P_{efflux}} \text{ is controlled by p-ACA.} \end{cases}$$

PapA, PapB, and PapC mRNAs are modeled separately because their translation rates are variable, even though they are expressed from the same promoter [7].

The protein folding process for the 4 enzymes and the efflux pump is modeled explicitly. Additionally, the enzyme deaminase is expressed constitutively but its concentration rises to steady state and thus its dynamics are also modeled. The enzyme mass balance equations are

$$\begin{aligned}
\frac{\mathrm{dpapA_{uf}}}{\mathrm{d}t} &= v(\mathrm{mRNA_{papA}}) - \eta\cdot\tau\cdot\mathrm{papA_{uf}} - \delta\cdot\mathrm{papA_{uf}} - \rho\cdot\tau\cdot\mathrm{papA_{uf}},\\
\frac{\mathrm{dpapB_{uf}}}{\mathrm{d}t} &= v(\mathrm{mRNA_{papB}}) - \eta\cdot\tau\cdot\mathrm{papB_{uf}} - \delta\cdot\mathrm{papB_{uf}} - \rho\cdot\tau\cdot\mathrm{papB_{uf}},\\
\frac{\mathrm{dpapC_{uf}}}{\mathrm{d}t} &= v(\mathrm{mRNA_{papC}}) - \eta\cdot\tau\cdot\mathrm{papC_{uf}} - \delta\cdot\mathrm{papC_{uf}} - \rho\cdot\tau\cdot\mathrm{papC_{uf}},\\
\frac{\mathrm{dLAAO_{uf}}}{\mathrm{d}t} &= v(\mathrm{mRNA_{LAAO}}) - \eta\cdot\tau\cdot\mathrm{LAAO_{uf}} - \delta\cdot\mathrm{LAAO_{uf}} - \rho\cdot\tau\cdot\mathrm{LAAO_{uf}},\\
\frac{\mathrm{dP_{efflux_{uf}}}}{\mathrm{d}t} &= v(\mathrm{mRNA_{P_{efflux_{uf}}}}) - \eta\cdot\tau\cdot\mathrm{P_{efflux_{uf}}} - \delta\cdot\mathrm{P_{efflux_{uf}}} - \rho\cdot\tau\cdot\mathrm{P_{efflux_{uf}}},\\
\frac{\mathrm{dpapA}}{\mathrm{d}t} &= \eta\cdot\tau\cdot\mathrm{papA_{uf}} - \delta\cdot\mathrm{papA} - \rho\cdot\tau\cdot\mathrm{papA},\\
\frac{\mathrm{dpapB}}{\mathrm{d}t} &= \eta\cdot\tau\cdot\mathrm{papB_{uf}} - \delta\cdot\mathrm{papB} - \rho\cdot\tau\cdot\mathrm{papB},\\
\frac{\mathrm{dpapC}}{\mathrm{d}t} &= \eta\cdot\tau\cdot\mathrm{papC_{uf}} - \delta\cdot\mathrm{papC} - \rho\cdot\tau\cdot\mathrm{papC},\\
\frac{\mathrm{dLAAO}}{\mathrm{d}t} &= \eta\cdot\tau\cdot\mathrm{LAAO_{uf}} - \delta\cdot\mathrm{LAAO} - \rho\cdot\tau\cdot\mathrm{LAAO},\\
\frac{\mathrm{dP_{efflux}}}{\mathrm{d}t} &= \eta\cdot\tau\cdot\mathrm{P_{efflux_{uf}}} - \delta\cdot\mathrm{P_{efflux}} - \rho\cdot\tau\cdot\mathrm{P_{efflux}},\\
\frac{\mathrm{ddeaminase}}{\mathrm{d}t} &= V(\mathrm{deaminase}) - \delta\cdot\mathrm{deaminase}.
\end{aligned} \tag{2.6}$$

The constant ρ is the protein degradation rate and η is the protein folding rate. The function v is the translation rate equation:

$$v(m) = \frac{m}{\mathrm{T_{init}} + \frac{\mathrm{L}_m}{\mathrm{R}}}, \tag{2.7}$$

where $\mathrm{T_{init}}$ is the transcription initiation rate, L_m is the length of the mRNA, and R is the ribosome elongation rate. All fixed parameters are defined in [Table 2.2]. Some parts of the model deviate from both Merzbacher et al. [7] and Carothers and Stevens [9]. $\text{V}_{\text{chorismate}}$ is reduced from 1100 1/s to 150 1/s to dampen metabolite overshoot, while still being within the range of chorismate production in *E. coli* [9]. PapA, PapB, and PapC catalyzed biosynthesis rate use *mm* instead of *f* to improve product throughput.

Parameter	Symbol	Value	Units
Chorismate production rate	$\text{V}_{\text{chorismate}}$	150	1/s
Deaminase production rate	$\text{V}_{\text{deaminase}}$	10	1/s
Translation initiation rate	T_{init}	0.2	1/s
p-AF loss	L	$1.40 \cdot 10^{-5}$	1/s
mRNA degradation rate	M	0.003	1/s
Protein degradation rate	P	0.0002	1/s
Protein folding rate	F	2	1/s
Dilution rate	δ	0.000579	1/s
DNA duplication rate	β	0.000578	1/s
Avogadro number	N_{A}	$6.02 \cdot 10^{23}$	NA
Cell volume	Vol_{cell}	$2.50 \cdot 10^{-17}$	L
Toxicity constant	k_i	$5.00 \cdot 10^{-5}$	M
Metabolite induced toxicity	t_a	0.0005	NA
Protein induced toxicity	t_p	50	NA
Enzyme induced toxicity	t_l	50	NA
Pap operon mRNA length	L_m	3400	nucleotides
Efflux pump mRNA length	L_m	2900	nucleotides
LAAO mRNA length	L_m	1600	nucleotides
Ribosome elongation rate	R	20	amino acids/s
Deaminase k_{cat}	k_{cat}	5	M/s
Deaminase k_{m}	k_{m}	$1.00 \cdot 10^{-6}$	M
PapA k_{cat}	k_{cat}	0.2975	M/s
PapA k_{m}	k_{m}	0.056	M
PapB k_{cat}	k_{cat}	39	M/s
PapB k_{m}	k_{m}	0.38	M
PapC k_{cat}	k_{cat}	20.44	M/s
PapC k_{m}	k_{m}	0.555	M
LAAO k_{cat}	k_{cat}	1.29	M/s
LAAO k_{m}	k_{m}	10.82	M
Efflux pump rate	k_{m}	50	M

Table 2.2: P-Aminostyrene model kinetic parameters

2.2 BayesOpt Loop

Once the model was constructed, we selected the continuous and discrete decision variables in [Table 2.1] as our optimizable parameters, where the continuous decision variables are bound between $5 \cdot 10^{-6}$ and $1 \cdot 10^{-4}$, based on the ranges given by the work from Carothers and Stevens [9]. This defines the search space. We chose to use BayesOpt as our optimizer for the reasons detailed in the previous chapter. In addition to these reasons, previous work on the p-AS model used BayesOpt and was successful in optimizing their parameters [7]. As per [Table 2.1], all initial conditions are set to 0mM.

The acquisition function deployed in our imported implementation of BayesOpt is the Tree Parzen Estimator, from the TreeParzen.jl package [22], which models the probability of observing a set of parameters given an objective value [16]. It is important that the model is not reliant on the shape of the probability distribution from which the data was drawn, as we cannot assume the p-AS pathway parameter space's shape or convexity [7].

To optimize both the continuous and discrete decision variables, the BayesOpt loop chooses a set of parameter values, and solves the ODE model. The solution is checked for constraint violation, (if applicable) and the objective value is returned. This loop is run 1000 times, choosing new parameter values based on the highest expected improvement of the objective [15]. The lowest objective value, along with its parameters, are returned as the final solution.

2.3 Objective Function

The objective function serves as an input for BayesOpt to efficiently search the design space to find architecture types and kinetic parameters that return the lowest values [6, 7]. In the context of metabolic engineering, it is practical to define the objective function as the difference between the total product produced of the cell and the cost incurred by the cell by making the product. The term production is fairly straightforward to define in metabolic engineering, but cost can include a variety of different variables at different scales and units, including toxicity, genetic instability, and the energy and resources required to produce the ribosomes, cofactors, and pathway intermediates for the product. Our objective function also includes the newly defined total proteome and proteome rate constraints, ($\mathrm{C}_{\text{total proteome}}$ and $\mathrm{C}_{\text{proteome rate}}$) and the complexity regularization term ($\mathrm{R}_{\text{complexity}}$). The overall objective function can be simplified to:

$$J = \alpha J_{\text{prod}} + (1-\alpha)J_{\text{cost}} + \mathrm{C}_{\text{total proteome}} + \mathrm{C}_{\text{proteome rate}} + \mathrm{R}_{\text{complexity}} \tag{2.8}$$

Where the minimization of J_{prod} represents the maximization of product flux, and J_{cost} penalizes the total amount of enzyme expressed [7]. α represents the relative weight to balance the two performance objectives as described in section 2.6. A new J_{prod} is defined as:

$$J_{\text{prod}} = \frac{1}{\int_0^T f(\mathrm{P}_{\text{efflux}}, \text{p-ACA})\, dt} \tag{2.9}$$

Where $f(e,x)$ is the equation in [Equation 2.3]. The original J_{prod} took the difference between chorismate production and p-AS production to minimize the flux between these two values. An unintended consequence of this definition of J_{prod} was that it encouraged an overshoot of chorismate over the production of p-AS. The new definition simply takes the reciprocal of p-AS production over the time period to convert J_{prod} to a minimization problem. J_{cost} is implemented as defined by Merzbacher et al. [7]:

$$
\begin{aligned}
J_{\text{cost}} = \int_0^T \{ & \omega_1 u(\text{p-AF}, k_{1,\,\text{p-AF}}, \theta_{1,\,\text{p-AF}}) + (1-\omega_1)u(\text{p-ACA}, k_{1,\,\text{p-ACA}}, \theta_{1,\,\text{p-ACA}}) \\
& + \omega_1 u(\text{p-AF}, k_{2,\,\text{p-AF}}, \theta_{2,\,\text{p-AF}}) + (1-\omega_1)u(\text{p-ACA}, k_{2,\,\text{p-ACA}}, \theta_{2,\,\text{p-ACA}}) \\
& + \omega_1 u(\text{p-AF}, k_{3,\,\text{p-AF}}, \theta_{3,\,\text{p-AF}}) + (1-\omega_1)u(\text{p-ACA}, k_{3,p-ACA}, \theta_{3,\,\text{p-ACA}}) \\
& + \omega_2 u(\text{p-AF}, k_{4,\,\text{p-AF}}, \theta_{4,\,\text{p-AF}}) + (1-\omega_2)u(\text{p-ACA}, k_{4,\,\text{p-ACA}}, \theta_{4,\,\text{p-ACA}}) \\
& \quad + \omega_3 u(\text{p-AF}, k_{5,\,\text{p-AF}}, \theta_{5,\,\text{p-AF}}) + (1-\omega_3)u(\text{p-ACA}, k_{5,\,\text{p-ACA}}, \theta_{5,\,\text{p-ACA}}) \}\mathrm{d}t,
\end{aligned}
\tag{2.10}
$$

where pathway cost is the sum of all heterologous enzyme transcription rates across all loci and ligands.

2.4 Total Proteome and Proteome Rate Constraints

Proteomic constraints were added to the model to represent the physical limitations of a cell's growth. There is a limit to both the total protein in a cell and the rate at which a cell can produce those proteins because the resources to create and transport those proteins (nucleotides, RNA polymerases, Transfer RNA (tRNA)s, ribosomes, permeases, etc.) get exhausted [23]. If the decision variables are given a set of values and the model simulates the production of more protein in the pathway than is possible for a cell, the architecture should be marked as invalid. Total proteome constraint and proteome rate constraint are defined as:

$$
\begin{aligned}
\mathrm{C}_{\text{total proteome}} &= \begin{cases} 0 & \text{if: } \Phi_{\mathrm{p}} \le \mathrm{c}_{\mathrm{p}}, \text{ for all time slices,} \\ 1.00 \cdot 10^8 & \text{otherwise.} \end{cases} \\
\mathrm{C}_{\text{proteome rate}} &= \begin{cases} 0 & \text{if: } \Phi_{\mathrm{pr}} \le \mathrm{c}_{\mathrm{pr}}, \\ 1.00 \cdot 10^8 & \text{otherwise.} \end{cases}
\end{aligned}
\tag{2.11}
$$

The constants c_{p} and c_{pr} are the assigned constraint thresholds to total proteome and proteome rate, respectively. Φ_{p} is the total protein produced from the p-AS pathway at one time slice,

$$
\Phi_{\mathrm{p}} = \mathrm{papA} + \mathrm{papB} + \mathrm{papC} + \mathrm{LAAO} + \mathrm{P}_{\text{efflux}},
\tag{2.12}
$$

and Φ_{pr} is the total mRNA produced from the p-AS pathway. Because the proteome rate inherently depends on the previous time slice, Φ_{pr} is described as pesudocode:

```
$\Phi_{\mathrm{pr}} \leftarrow 0$
for time slice in time slices do
    $\Phi_{\mathrm{pr}}$ += $\max(\mathrm{mRNA}_{\mathrm{papA}}(\text{time slice}) - \mathrm{mRNA}_{\mathrm{papA}}(\text{time slice}-1), 0)$
    $+\max(\mathrm{mRNA}_{\mathrm{papB}}(\text{time slice}) - \mathrm{mRNA}_{\mathrm{papC}}(\text{time slice}-1), 0)$
    $+\max(\mathrm{mRNA}_{\mathrm{papC}}(\text{time slice}) - \mathrm{mRNA}_{\mathrm{LAAO}}(\text{time slice}-1), 0)$
    $+\max(\mathrm{mRNA}_{\mathrm{LAAO}}(\text{time slice}) - \mathrm{mRNA}_{\mathrm{LAAO}}(\text{time slice}-1), 0)$
    $+\max(\mathrm{mRNA}_{\mathrm{P}_{\text{efflux}}}(\text{time slice}) - \mathrm{mRNA}_{\mathrm{P}_{\text{efflux}}}(\text{time slice}-1), 0)$
end for
```

This pseudocode adds any positive rates of change in mRNA of each of the enzymes in the pathway across the model's runtime. Negative values are excluded to prevent high proteome rate values from being lowered below c_{pr}.

For each value of c_p and c_{pr} swept, the BayesOpt loop was run 100 times to find the percentage of BayesOpt loop runs that satisfy the constraint, and the number of unique architectures that find the lowest objective in a BayesOpt run. The dynamics of optimal architecture distributions and the objective minimum for each constraint were visualized.

2.5 Complexity Regularization

Including a regularization term for complexity provides a means to increase the objective value for architectures by how expensive they would be to engineer relative to each other, thereby punishing difficult and costly-to-implement circuits. We define complexity as:

$$\mathrm{R}_{\mathrm{complexity}} = \lambda \cdot (\mathrm{loops} + 2 \cdot \mathrm{biosensors}), \tag{2.13}$$

where λ is the complexity weight, a loop is an engineered transcription factor which activates or represses the pathway enzymes (PapABC, LAAO, or the efflux pump), and a biosensor is the engineered metabolite sensing of p-AF or p-ACA [7, 9]. Each loop and biosensor in the circuit adds additional logic that researchers must engineer into the microbial system. Biosensors are more complex than loops, so we double their weighting as a basic heuristic. This value is added to the objective function, affecting the viability of more complex architectures when $\lambda > 0$. To measure the effect of complexity regularization on architecture distribution, the BayesOpt loop was run 100 times at each λ from 0 until the only viable architecture was the “No Control” architecture, $5 \cdot 10^{-1}$. The results were then visualized.

2.6 Objective Tuning and Normalization

J_{prod} and J_{cost} were normalized using the normalization formula ($\frac{x-xmin}{xmax-xmin}$, where *xmax* and *xmin* are constants restricting the lower and upper bounds, respectively). α values were then swept and J_{prod} and J_{cost} objective values were recorded [Figure 2.2]. Generally, as α increased from 0.0 to 1.0, J_{prod} becomes a higher proportion of the total weight of the two normalized objectives, and J_{cost} becomes a lower proportion of the total weight. To ensure equal weighting between production and cost objectives, α was set to 0.65 (where the normalized production and cost objectives are roughly equal in value) under total proteome, proteome rate, and complexity experiments.

2.7 ODE Solver Benchmarking

To measure the efficiency of different ODE solvers implemented in Julia relative to the Livermore Solver for Ordinary Differential Equations (LSODA), available in Python, each function was run and timed 25 times under the same initial conditions and input parameters, and the average was recorded. The average runtime for the Python ODE solver LSODA was 0.1102 seconds, a significant increase compared to the average runtime for the Julia ODE solvers RosenBrock23 (a Kaps–Rentrop method), TRBDF2 (a Runge-Kutta method), Quasi-constant timestep Numerical Differentiation Formula (QNDF), and (Fixed-leading coefficient Backward Differentiation Formula) (FBDF), at 0.0626,

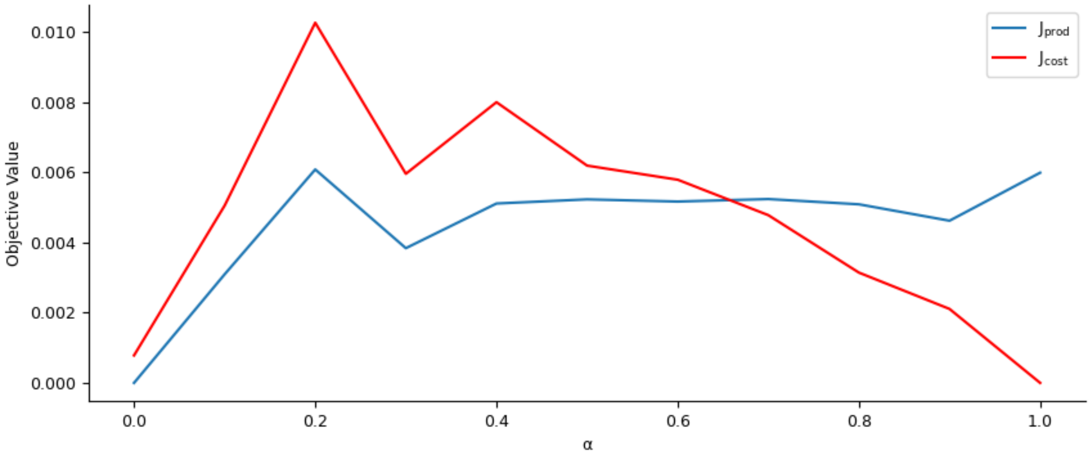

Figure 2.2: **Normalized production (J_{prod}) and cost (J_{cost}) objective values by α values.** As α increases, J_{prod} becomes a higher proportion of the total objective function, while J_{cost} becomes a lower proportion of the total weight. At $\alpha = 0.65$, the normalized production and cost objectives are roughly equal in value, at 0.00465.

0.0262, 0.0238, 0.0252, respectively [Figure 2.3]. All 4 Julia solvers demonstrate a significant decrease in runtime compared to LSODA.

The robustness of RosenBrock23 relative to LSODA was measured by running the ODE with the same parameters 8 times, and comparing the model dynamics. In all 8 runs of LSODA, there were drastic fluctuations in value for LAAO concentrations that were not reflected in the set of ODEs. These fluctuations did not occur with RosenBrock23.

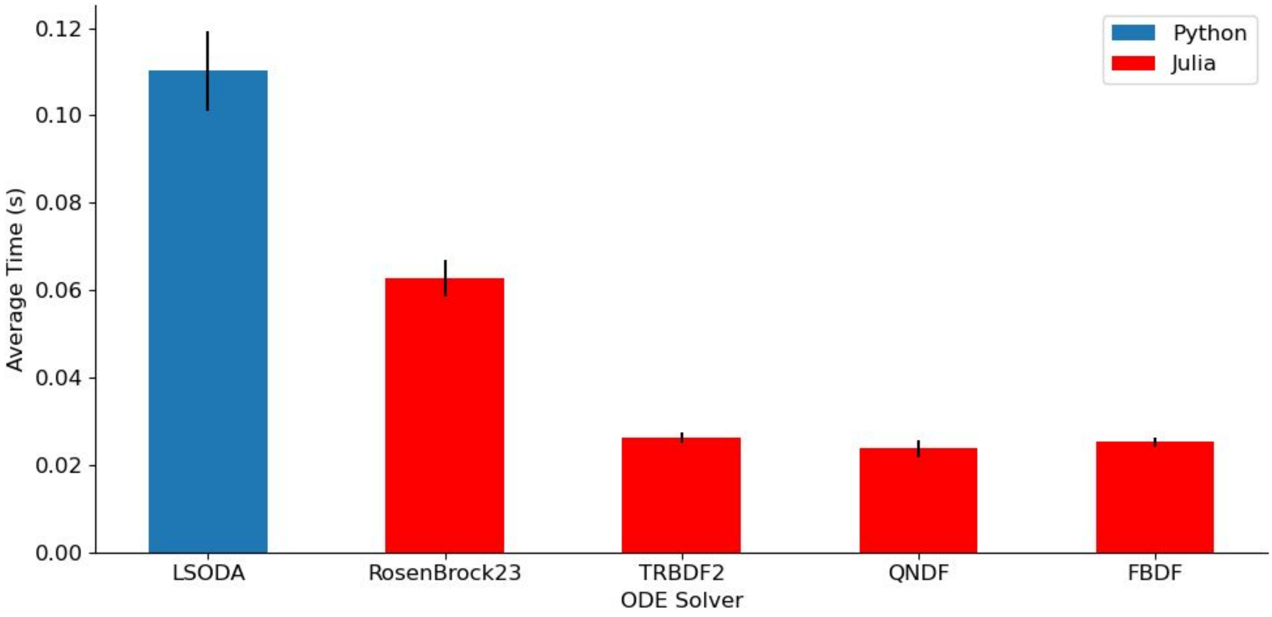

Figure 2.3: **Runtime benchmarking of ODE solvers implemented in Julia and Python.** Average runtimes of Python (blue) and Julia (red) solvers across 25 ODE solution calculations.

Chapter 3

Results

3.1 Total Proteome Constraint

We analyze the effect of the total proteome constraint threshold (c_p) on the found optimal architectures in BayesOpt runs by sweeping c_p from $1 \cdot 10^{-6}$ to $5 \cdot 10^{-5}$. Within this range, as c_p is relaxed, a higher percentage of BayesOpt runs find a solution which satisfies the constraint until $3 \cdot 10^{-5}$, where it reaches 100% [Figure 3.1a]. c_p was then swept from $2 \cdot 10^{-6}$ to $1 \cdot 10^{-3}$ to visualize the change in number of unique optimal architectures [Figure 3.1b]. These data suggest that at very tight total proteome constraints, the lower percentage of runs that find an optimal architecture leads to fewer unique optimal architectures. As the constraint loosens, there are a higher percentage of runs that find optimal architectures, leading to more unique optimal architectures. As the constraint loosens further, the architectures that produce the best product flux dominate as the most optimal architectures, reducing the number of unique optimal architectures. The minimum objective value found for a selection of BayesOpt runs at certain c_p values are graphed [Figure 3.1c], and the distribution of unique optimal architectures for those runs are displayed [Figure 3.1d]. The minimum objective value found on a given BayesOpt run decreases as the constraint value increases. At $1 \cdot 10^{-6}$, the constraint is so prohibitive, none of the BayesOpt runs find a solution which satisfies a constraint. As c_p decreases, the distribution of architectures changes from the optimal architectures of weak ($5 \cdot 10^{-5}$) and no (Control) constraints, and favors a wider distribution of architectures. At $c_p = 2 \cdot 10^{-6}$, architecture diversity falls as only a few architectures can satisfy such a low constraint, including (N, $R_{\text{p-ACA}}$, $A_{\text{p-ACA}}$), which has the highest share of optimal architectures (note that each architecture is visualized in [Figure 2.1]). We conclude that the total proteome constraint filters out architecture and kinetic parameter sets that have better objective values through improved product flux, but violate the model's definition of the maximum concentration of protein possible in the cell.

3.2 Proteome Rate Constraint

We now analyze the effect of the proteome rate constraint threshold (c_{pr}) on the found optimal architectures in BayesOpt runs by sweeping c_{pr} from $5 \cdot 10^{-7}$ to $7.5 \cdot 10^{-6}$. Within this range, as c_{pr} is relaxed, a higher percentage of BayesOpt runs find a solution which satisfies the constraint until $5 \cdot 10^{-6}$, where it reaches 100% [Figure 3.2a]. c_{pr} was then swept from $5 \cdot 10^{-8}$ to $4 \cdot 10^{-5}$ to visualize the change in number of unique optimal architectures [Figure 3.2b]. Similar to the results from the

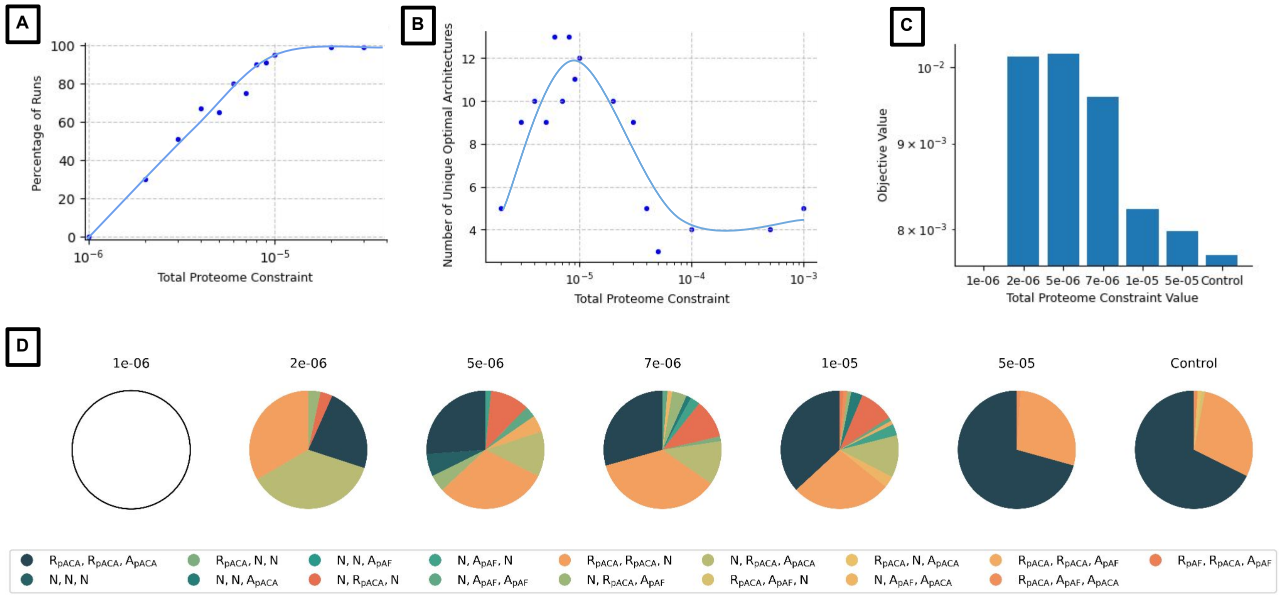

Figure 3.1: **Total proteome constraint experiments.** The trend curves in a and b were added manually and act as visual aids. For each total proteome constraint value, 100 BayesOpt runs of 1000 iterations are run. a: Percentage of BayesOpt runs that find an optimal solution by total proteome constraint. b: Number of unique optimal architectures for each total proteome constraint. c: Minimum objective value for total proteome constraint BayesOpt runs of interest. d: Architecture distributions for total proteome constraint BayesOpt runs of interest.

total proteome constraint experiment, at very tight proteome rate constraints, the lower percentage of runs that find an optimal architecture leads to fewer unique optimal architectures. As the constraint loosens, there are a higher percentage of runs that find optimal architectures, leading to more unique optimal architectures. As the constraint loosens further, the architectures that produce the best product flux dominate as the most optimal architectures, reducing the number of unique optimal architectures. The minimum objective value found for a selection of BayesOpt runs at certain c_{pr} values are graphed [Figure 3.2c], and the distribution of unique optimal architectures for those runs are displayed [Figure 3.2d]. The minimum objective value found on a given BayesOpt run decreases as the constraint value increases, with the exception of $c_{pr} = 5 \cdot 10^{-8}$, where the minimum objective value is 0.0101, which is lower than the minimum objective value for the constraint of $1 \cdot 10^{-7}$ at 0.0110, despite the constraint being less restrictive. This is a surprising result, but we hypothesize that the reduced percentage of runs made the minima of these two constraints more variable than the less restrictive constraints. At $c_{pr} = 1 \cdot 10^{-8}$, the constraint is so prohibitive, none of the BayesOpt runs find a solution which satisfies a constraint. As c_{pr} decreases, the distribution of architectures changes from the optimal architectures of weak ($1 \cdot 10^{-6}$) and no (Control) constraints, and favors a wider distribution of architectures. At $2 \cdot 10^{-6}$, architecture diversity falls as only a few architectures can satisfy such a low constraint. Similar to the total proteome constraint, the proteome rate constraint filters out architecture and kinetic parameter sets that have better objective values through improved product flux, but violate our model's definition of the maximum rate at which protein can be produced in the cell.

3.3 Regularizing for Complexity

Finally, we analyze the effect of $R_{\text{complexity}}$ on found optimal architectures in BayesOpt runs by sweeping λ from 0 to $3 \cdot 10^{-1}$. We define average complexity as the weighted average of the complexity of each architecture in the distribution. We find that average complexity decreases from 6.222 at $\lambda = 0$, to 0 at $\lambda = 3 \cdot 10^{-1}$ [Figure 3.3]. Between the two values, the distribution of architectures gradually changes from favoring more complex architectures ($R_{\text{p-ACA}}$, $R_{\text{p-ACA}}$, N, and $R_{\text{p-ACA}}$, $R_{\text{p-ACA}}$, $A_{\text{p-ACA}}$, at complexity values of 4 and 7, respectively) towards less complex architectures ($R_{\text{p-ACA}}$, N, N, and N, N, N, at complexity values of 3 and 0, respectively). The less complex architectures have fewer loops and sensors than more complex architectures, reducing their ability to dynamically regulate pathway enzyme concentrations. While some of the more complex architectures result in higher product flux, they inherently have a higher monetary cost to engineer *in vitro*. By increasing λ, the relative importance of this cost is increased until the no control architecture is the only architecture that finds the lowest objective value in any given BayesOpt run.

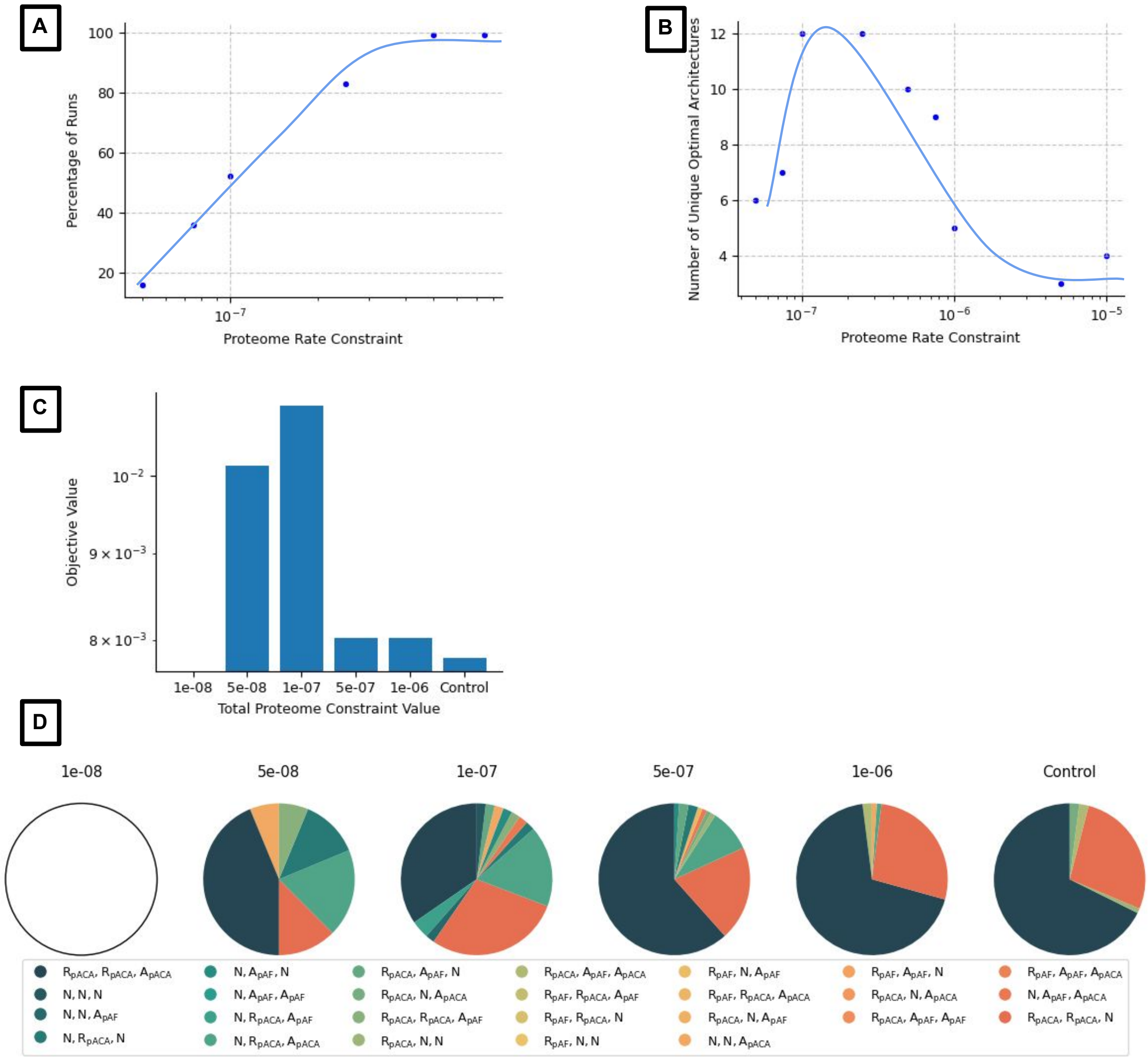

Figure 3.2: **Proteome rate constraint experiments.** The trend curves in a and b were added manually and act as visual aids. For each proteome rate constraint value, 100 BayesOpt runs of 1000 iterations are run. a: Percentage of BayesOpt runs that find an optimal solution by proteome rate constraint. b: Number of unique optimal architectures for each proteome rate constraint. c: Minimum objective value for proteome rate constraint BayesOpt runs of interest. d: Architecture distributions for proteome rate constraint BayesOpt runs of interest.

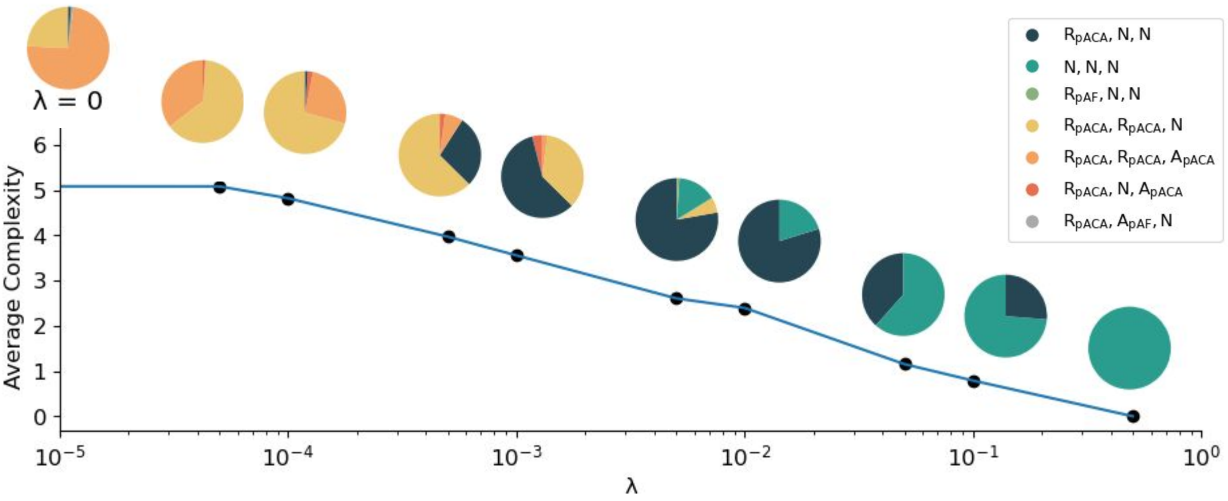

Figure 3.3: **Architecture distribution and the average complexity by complexity weight (λ).** Average complexity is calculated by taking the weighted average of the complexity of each architecture in the distribution. For each value of λ, 100 BayesOpt runs of 1000 iterations are performed. The average complexity of $\lambda = 0$ is 6.222, which cannot be displayed on a logaritmic-scale graph.

Chapter 4

Discussion

Synthetic biology allows scientists to design complex circuits across multiple scales to perform a specific biological function [12]. The design of these circuits can incur unnecessary monetary costs if multiple circuit architectures must be designed [6]. Computationally modeling the pathway of interest where the circuit will function provides an opportunity to filter for architectures which perform well under certain objectives and constraints, but the parameter search space of large metabolic pathways is prohibitively expensive to search using brute-force methods and many common optimization algorithms [15, 22, 24]. In this work, we implemented a model of the p-AS pathway in Julia for improved runtime and robustness compared to Python. A BayesOpt loop was implemented to search the large parameter space for high expected improvement, including a definition of J_{cost} which further favors p-AS production. Two constraints and one regularization term, $\text{C}_{\text{total proteome}}$, $\text{C}_{\text{proteome rate}}$, and $\text{R}_{\text{complexity}}$ were implemented and shown to change the optimal architecture distribution to adhere to the applied constraints and regularization.

A circuit architecture becomes more expensive to design *in vitro* as its complexity increases [6]. $\text{R}_{\text{complexity}}$ is designed with the subjective nature of balancing engineering costs and maximum product flux in mind. λ can be adjusted as necessary to find an architecture that provides the best product flux at a given maximum complexity to design the most cost-effective circuit possible.

This work is relevant to product maximization of other metabolic pathways, particularly to those across multiple scales, where ODE models are numerically stiff. The creation of the BayesOpt loop and constraint definitions remain consistent across pathways, meaning the additional work required to find optimal architectures for other pathways is limited to the definition of a set of ODEs, production and cost objectives, and potentially a complexity function, depending on the circuit of interest.

There are many additional considerations for adjusting the BayesOpt loop. For example, the α weighting could be adjusted to favor production or cost, which are predicted to favor kinetic parameter values that increase product flux or reduce enzyme expression, respectively. The initial conditions could be nonzero and nonuniform, which would more accurately represent a microbial factory. Architectures of interest could be selected at a bias to encourage BayesOpt to explore more sets of kinetic parameters on them.

There are more production objectives that could be implemented to create a more complete J_{prod}, such as explicit identifications of rise time, overshoot, and steady-state titer. Additional constraints could also be implemented to more accurately represent the cell, such as maximum ATP production and maximum pathway intermediates. The implementation of a maximum pathway intermediates constraint is particularly noteworthy given the unrealistic overshoot of chorismate production returned by the model for optimal architectures and kinetic parameters (an overshoot of 10^{13} mM). A constraint on chorismate production is hypothesized to improve model accuracy by removing kinetic parameter solutions that lead to such high chorismate production values. This work looked at $C_{\text{total proteome}}$, $C_{\text{proteome rate}}$, and $R_{\text{complexity}}$ individually. Further work on this topic could include combining the constraints and regularization term to find a subset of candidate architectures to be engineered *in vitro*.

Word Count: 4492

Bibliography

(1) Wuest, D.; Hou, S.; Lee, K. *Comprehensive Biotechnology* **2011**, 617–628.

(2) Khan, S.; Ullah, M. W.; Siddique, R.; Nabi, G.; Manan, S.; Yousaf, M.; Hou, H. *International Journal of Genomics* **2016**, *2016*, 1–14.

(3) Kobayashi, H.; Kærn, M.; Araki, M.; Chung, K.; Gardner, T. S.; Cantor, C. R.; Collins, J. J. *Proceedings of the National Academy of Sciences* **2004**, *101*, 8414–8419.

(4) Ni, C.; Dinh, C. V.; Prather, K. L. *Annual Review of Chemical and Biomolecular Engineering* **2021**, *12*, 519–541.

(5) Blazeck, J.; Alper, H. *Biotechnology Journal* **2010**, *5*, 647–659.

(6) Hiscock, T. W. *BMC Bioinformatics* **2019**, *20*, DOI: 10.1186/s12859-019-2788-3.

(7) Merzbacher, C.; Mac Aodha, O.; Oyarzún, D. A. *ACS Synthetic Biology* **2023**, *12*, PMID: 37339382, 2073–2082.

(8) Carothers, J. M.; Goler, J. A.; Juminaga, D.; Keasling, J. D. *Science* **2011**, *334*, 1716–1719.

(9) Stevens, J. T.; Carothers, J. M. *ACS Synthetic Biology* **2015**, *4*, PMID: 25314371, 107–115.

(10) Liu, F.; Heiner, M.; Gilbert, D. *Briefings in Bioinformatics* **2022**, *23*, DOI: 10.1093/bib/bbac081.

(11) Wooley, J. C.; Lin, H., *Catalyzing inquiry at the interface of Computing and Biology*; National Academies Press: 2005.

(12) Southern, J.; Pitt-Francis, J.; Whiteley, J.; Stokeley, D.; Kobashi, H.; Nobes, R.; Kadooka, Y.; Gavaghan, D. *Progress in Biophysics and Molecular Biology* **2008**, *96*, Cardiovascular Physiome, 60–89.

(13) Wanner, G.; Hairer, E., *Solving ordinary differential equations II*; Springer Berlin Heidelberg: 1996.

(14) Frank, S. A. *Frontiers in Ecology and Evolution* **2022**, *10*, DOI: 10.3389/fevo.2022.1010278.

(15) Frazier, P. I. A Tutorial on Bayesian Optimization, 2018.

(16) Bergstra, J.; Bardenet, R.; Bengio, Y.; Kégl, B. In *Advances in Neural Information Processing Systems*, ed. by Shawe-Taylor, J.; Zemel, R.; Bartlett, P.; Pereira, F.; Weinberger, K., Curran Associates, Inc.: 2011; Vol. 24.

(17) Bashor, C. J.; Collins, J. J. *Annual Review of Biophysics* **2018**, *47*, 399–423.

(18) Winkler, J. D.; Halweg-Edwards, A. L.; Gill, R. T. *Metabolic Engineering Communications* **2016**, *3*, 227–233.

(19) Lüders, C.; Sturm, T.; Radulescu, O. *Bioinformatics Advances* **2022**, *2*, vbac027.

(20) Dromms, R. A.; Styczynski, M. P. *Metabolites* **2012**, *2*, 1090–1122.

(21) Gratie, D.-E.; Iancu, B.; Petre, I. In *Formal Methods for Dynamical Systems: 13th International School on Formal Methods for the Design of Computer, Communication, and Software Systems, SFM 2013, Bertinoro, Italy, June 17-22, 2013. Advanced Lectures*, Bernardo, M., de Vink, E.,

Di Pierro, A., Wiklicky, H., Eds.; Springer Berlin Heidelberg: Berlin, Heidelberg, 2013, pp 29–62.

(22) Bergstra, J.; Yamins, D.; Cox, D. In *Proceedings of the 30th International Conference on Machine Learning*, ed. by Dasgupta, S.; McAllester, D., PMLR: Atlanta, Georgia, USA, 2013; Vol. 28, pp 115–123.

(23) Stoebel, D. M.; Dean, A. M.; Dykhuizen, D. E. *Genetics* **2008**, *178*, 1653–1660.

(24) Goikhman, M. Y.; Yevlampieva, N. P.; Kamanina, N. V.; Podeshvo, I. V.; Gofman, I. V.; Mil'tsov, S. A.; Khurchak, A. P.; Yakimanskii, A. V. *Polymer Science Series A* **2011**, *53*, 457–468.

Appendix: Project Code

Please find the associated code for the BayesOpt loop and p-AS model implementation at:

`https://github.com/ngoguened/2024_Project_Code_Edinburgh`