

**APPLICATION OF CONSTRAINT-BASED MODELING
FOR *IN SILICO* ANALYSIS AND PREDICTION
OF CHO CELL GLYCOSYLATION**

by

Michaela Gallucci

A thesis submitted to the Faculty of the University of Delaware in partial fulfillment of the requirements for the degree of Master in Chemical Engineering

Spring 2018

© 2018 Michaela Gallucci
All Rights Reserved

**APPLICATION OF CONSTRAINT-BASED MODELING
FOR *IN SILICO* ANALYSIS AND PREDICTION
OF CHO CELL GLYCOSYLATION**

by

Michaela Gallucci

Approved: _____
Kelvin H. Lee, Ph.D.
Professor in charge of thesis on behalf of the Advisory Committee

Approved: _____
Eric M. Furst, Ph.D.
Chair of the Department of Chemical and Biomolecular Engineering

Approved: _____
Babatunde A. Ogunnaike, Ph.D.
Dean of the College of Engineering

Approved: _____
Ann L. Ardis, Ph.D.
Senior Vice Provost for Graduate and Professional Education

ACKNOWLEDGMENTS

Oftentimes scientists are placed solely in the context of their work, known only for innovative creations and landmark discoveries. However, my journey of discovery in science has been directly tied to my discovery of self, and it is impossible to decouple the two. This thesis is a manifestation of both my growth as a scientist and my growth as a person during my thesis and gender transition. As such, many of the marvelous individuals mentioned in this thesis are not thanked simply for their contributions to my science – but also to providing me with the personal environment I needed to succeed.

First and foremost, I would like to thank my advisor (Dr. Kelvin Lee) and committee (Dr. Abraham Lenhoff and Dr. Millicent Sullivan) for their scientific guidance and personal support. I did not think it was possible to learn as much as I did in the span of 2 years, but I look back on my time in the lab and am forever grateful for their wisdom, encouragement, and guidance.

Attempting to complete a Master's Thesis and a gender transition simultaneously has been one of the most difficult tasks of my life. However, the people I worked with on a daily basis not only made it achievable, they gave me a home where I could grow as both a scientist as a person. In my initial months as a graduate student, I must thank Ben Kremkow, Amalie Levy, and Jennifer Mantle for their guidance in writing, presentation skills, and the scientific principles that would shape my thesis. The second half of my time in the Lee lab was augmented greatly by both faculty (Leila Choe), and lab members: John Ruano-Salguero, Xiaolin Zhang,

Madolyn MacDonald, Jongyouon Baik, Jing Guo, Nathaniel Hamaker, Douglas Nmagu, Lie Min, Victoria Hunt, Alana Szkodny, and Will Hilliard. Their scientific expertise, writing skill, and wonderful companionship cultivated a professional, fun, and accepting environment for me to get the most out of my work and my time in the group. Finally, my friends across the ChE department (Ellie Oates, Caitlin Wood, Jon Lym, Camil Diaz, Joshua Condon, Eden Ford, and Julie Hipp) have helped make wonderful memories here I'll never forget.

In my personal life, I would be remiss if I did not dedicate a paragraph to my family. Their unwavering support and understanding during my time in graduate school and my transition have helped me rise to all the challenges I have faced. I would like to especially thank my sister, Jessica Gallucci, for the laughs, tears, support, and singing during this time – I don't know what I would do without you.

Lastly, my friends have been one of my biggest rocks throughout the challenges I have faced these past two years. Trudyann Buckley, one of my best friends, has been instrumental in my success for both my thesis and my transition. From the nights spent watching movies after my time at the hospital to spontaneous shopping trips to celebrate a new haircut, her presence has helped me in more ways than I can count. I only hope to be able to return a fraction of the strength you gave me someday. Though not physically in Delaware, this last paragraph must also go to my friends abroad: Mike Kritselis, Jake Puzycki, Kyle Lum, Durlav Mudbhari, Charlie Baldwin, Max Watkins, and Brandyn Bok. The hours I spent with them on the phone talking through my work, my transition, games, life, and everything in between have kept me strong and sane during the challenges faced these past few years. To all my family and friends – this thesis is as much yours as it is mine.

TABLE OF CONTENTS

LIST OF TABLES	ix
LIST OF FIGURES	x
ABSTRACT	xiii

Chapter

1	INTRODUCTION	1
1.1	The Importance of Chinese Hamster Ovary Cells in Biopharmaceutical Development	1
1.2	Glycosylation Alters Protein Therapeutic Performance	2
1.3	Mathematical Models Can Improve Glycosylation Engineering	4
1.4	Project Goals	6
1.5	Scope of Work	7
	REFERENCES	8
2	BACKGROUND OF GLYCOBIOLOGY AND MODELING TECHNIQUES	11
2.1	Introduction	11
2.2	The Process of N-Glycosylation	11
2.2.1	Glycan Synthesis and E.R. Transport	12
2.2.2	Intra-Golgi Protein Transport	14
2.2.3	Glycan Modification through Glycosyltransferase Processing ...	17
2.2.4	Metabolic Production of Sugar Co-substrates for Glycosylation	23
2.3	Overview of <i>In Silico</i> Glycosylation Modeling	29
2.3.1	First Attempts (1996-2005)	29
2.3.1.1	Shelikoff et al., 1996	29
2.3.1.2	Monica et al., 1997	32
2.3.1.3	Umaña et al., 1997	34
2.3.1.4	Kawano et al., 2005	37
2.3.1.5	Krambeck & Betenbaugh, 2005	39
2.3.2	Model Advancements (2006-2014)	42
2.3.2.1	Hossler et al., 2007	42
2.3.2.2	Kontoravdi et al., 2006	46

2.3.2.3	Senger & Karim, 2008.....	49
2.3.2.4	Krambeck et al., 2009.....	51
2.3.2.5	Jimenez et al., 2011	53
2.3.2.6	St. Amand et al., 2014	55
2.3.2.7	Jedrzejewski et al., 2014.....	59
2.3.3	Current State (2015-Present)	61
2.3.3.1	Spahn et al., 2015	61
2.3.3.2	Jimenez del Val. et al., 2016.....	64
2.4	Overview of Constraint-based Modeling	66
	REFERENCES	71
3	CONSTRAINT-BASED FRAMEWORK FOR GLYCOSYLATION MODELING.....	78
3.1	Introduction	78
3.2	Model Structure	79
3.3	Materials and Methods	83
	REFERENCES	85
4	ENZYMATIC ANALYSIS MODE	87
4.1	Introduction	87
4.2	Materials and Methods	87
4.2.1	Glycan Notation.....	87
4.2.2	Reaction Network Development	89
4.2.3	Implementation of Parameter-driven Glycosylation Modeling... ..	91
4.2.4	Fitting Algorithm Development	95
4.3	Results and Discussion	99
4.3.1	Sealover et al., 2013	100
4.3.2	Goh et al., 2014	104
4.3.3	Malphettes et al., 2010.....	108
4.3.4	Onitsuka et al., 2011	111
	REFERENCES	118
5	METABOLISM ANALYSIS MODE	121

5.1	Introduction	121
5.2	Materials and Methods	122
5.2.1	Reaction Network Development	122
5.2.2	Implementation of Parameter-driven Glycosylation Modeling	123
5.2.3	Fitting Algorithm Development	126
5.3	Results and Discussion	128
5.3.1	Wong et al., 2010.....	129
5.3.2	Gramer et al., 2011	132
5.3.3	St. Amand et al., 2014	137
5.3.4	Burleigh et al., 2011	140
	REFERENCES	148
6	DEVELOPMENT OF MATLAB GUI	151
6.1	Introduction	151
6.2	Features.....	151
	REFERENCES	156
7	CONCLUSIONS AND FUTURE WORK.....	157
7.1	Conclusions	157
7.2	Future Work.....	158
7.2.1	Improvement of Constraint-based Modeling for Glycosylation	159
7.2.2	Implementing Predictive Functions in Metabolism Analysis Mode.....	160
	REFERENCES	162
Appendix		
A	RAW MODEL OUTPUT (CHAPTER 4 RESULTS).....	163
A.1	Preface	163
A.2	Chapter 4.3.1 Raw Output	163
A.3	Chapter 4.3.2 Raw Output	165
A.4	Chapter 4.3.3 Raw Output	173
A.5	Chapter 4.3.4 Raw Output	176
B	RAW MODEL OUTPUT (CHAPTER 5 RESULTS).....	180

B.1	Preface	180
B.2	Chapter 5.3.1 ECP Values	181
B.3	Chapter 5.3.2 ECP Values	181
B.4	Chapter 5.3.3 ECP Values	182
B.5	Chapter 5.3.4 ECP Values	183
C	MATLAB SOURCE CODE.....	184
C.1	Preface	184
C.2	Source Code.....	184
C.2.1	GUI Interface (File – GlycoMapperExec.m).....	184
C.2.2	Enzyme Analysis Mode: Exploration Mode (File – MAIN_CRN_ExplorationMode.m).....	193
C.2.3	Enzyme Analysis Mode: Mapping Mode (Algorithm #1) (File – MAIN_CRN_MappingModeAlg1.m)	194
C.2.4	Enzyme Analysis Mode: Mapping Mode (Algorithm #2) (File – MAIN_CRN_MappingModeAlg2.m)	196
C.2.5	Enzyme Analysis Mode: Parameter Calibration (File – CRN_CalibrateParams.m)	197
C.2.6	Enzyme Analysis Mode: Upper and Lower Bound Generator (File – CRN_CreateConstraints.m)	199
C.2.7	Enzyme Analysis Mode: Outer Loop Management for Mapping Mode Alg2 (File – CRN_MetricOptimizer.m)	202
C.2.8	Enzyme Analysis Mode: Indexing Error Calculator (File – CRN_FindAdvancedError.m)	205
C.2.9	Enzyme Analysis Mode: Model Error Metric Calculator (File – CRN_FindAdvancedError2.m)	223
C.2.10	Enzyme Analysis Mode: Print Results (File – CRN_PrintResults.m)	238
C.2.11	Metabolism Analysis Mode: Parameter Calibration (File – MAIN_CCM_ExplorationMode.m)	240
C.2.12	Metabolism Analysis Mode: Mapping Mode (File – MAIN_CCM_MappingModeAlg1.m)	242
C.2.13	Metabolism Analysis Mode: Parameter Calibration (File – CCM_CalibrateParams.m)	243
C.2.14	Metabolism Analysis Mode: Generate Upper and Lower Bounds (File – CCM_CreateConstraints.m)	245
C.2.15	Metabolism Analysis Mode: ECP Calibration Utility Script (File – solveRecycle.m).....	249
C.2.16	Metabolism Analysis Mode: Print Results (File – CCM_PrintResults.m)	250

LIST OF TABLES

Table 2.1: Diagram of initial and final glycan as well as major glycosyltransferase function and substrate requirements (glycan images: Kim et al., 2009). 21	21
Table 4.1: List of ECP values and corresponding glycosyltransferase (or glycosyltransferase branch, terminated with “_B*”). 94	94
Table 4.2: List of all attributes used in calibration for Alg2 and the ECPs used to control each attribute. 99	99
Table A.1: (Chapter 4.3.1) Model output after calibration to control glycoform. 163	163
Table A.2: (Chapter 4.3.1) Model output after reducing the GnT1 ECP to zero. 165	165
Table A.3: (Chapter 4.3.2) Model output after calibration to control glycoform. 165	165
Table A.4: (Chapter 4.3.2) Model output after reducing the GnTI ECP to zero. 173	173
Table A.5: (Chapter 4.3.3) Model output after calibration to experimental glycoform. 173	173
Table A.6: (Chapter 4.3.3) Model output after reducing FucT ECP to zero. 175	175
Table A.7: (Chapter 4.3.4) Model output after calibration to control glycoform. 176	176
Table A.8: (Chapter 4.3.4) Model output after calibration to experimental glycoform (<i>ST6Gal1 overexpression</i>). 177	177
Table B.1: (Chapter 5.3.1) ECP values for all experimental conditions. 181	181
Table B.2: (Chapter 5.3.2) ECP values for all experimental conditions. 181	181
Table B.3: (Chapter 5.3.3) ECP values for all experimental conditions. 182	182
Table B.4: (Chapter 5.3.4) ECP values for all experimental conditions. 183	183

LIST OF FIGURES

Figure 2.1: Formation of $\text{Glc}_3\text{Man}_9\text{GlcNAc}_2\text{-P-P-Dol}$ in the E.R.....	13
Figure 2.2: Visual representation of protein transport via (A) vesicular transport or (B) cisternal maturation.....	15
Figure 2.3: (A) Visual representation of an unprocessed N- glycan upon exiting the E.R. (B) Visual representation of a terminally-processed N-glycan exiting the TGN.....	18
Figure 2.4: Approximate localization of major glycosyltransferases for N-glycosylation in the Golgi.	19
Figure 2.5: Metabolic network describing the conversion of simple sugars to co-substrates used in glycosylation (UDP-Gal, GDP-Fuc, GDP-Man, UDP-GNAc, CMP-NeuAc).....	25
Figure 2.6: Transformation of a physiological reaction network into a constraint-based model.	68
Figure 2.7: Illustration between the FBA and FVA algorithms for hypothetical solution space (shown in blue) for reactions v_1 and v_2 . Constraints within the solution space are represented as red dashed lines.....	70
Figure 3.1: Model assumptions used in both modes of operation using glycoform as only input.....	80
Figure 3.2: Organizational structure for programming and operating constraint-based model.....	81
Figure 3.3: (A) Discretization of reaction flux range (normalized to incoming M9 glycan flux) into ten ECPs. (B) Application of ECP for a single reaction converting M5 to M5(B1) via the enzyme GnTI. The plot on the right shows the final glycoform as a function of the ECP controlling GnTI, assuming all glycans started as M5 and only the reaction shown is considered.....	82
Figure 4.1: Glycan notation structure implemented in model. The colored circles correspond to the appropriately-colored substrings within the glycan string. Numbers in the string correspond to the number of appropriately-colored residues in the glycan.....	89
Figure 4.2: Sequence of glycosyltransferase action implemented by our model.	91

Figure 4.3: Workflow for Alg1. In Step 2, the <i>round()</i> function rounds the value in the parenthesis to the nearest integer. In Step 3, the <i>correctAlg()</i> function determines if the flux value inside the parenthesis exceeds the solution obtained via FBA in step 2. If so, it lowers the value in the parentheses to match this value.	95
Figure 4.4: Workflow for Alg2 parameter calibration.	98
Figure 4.5: Wild-type baseline glycoform (black dots), model glycoform prediction ranges (blue bars) and corresponding ECP values (table).....	101
Figure 4.6: Experimental glycoform (black dots), model prediction (blue bar), and corresponding ECP values (Table) after knockout of <i>Mgat1</i> (gene that codes for production of GnTI).....	103
Figure 4.7: Experimental baseline glycoform values (black dots), model prediction ranges (blue bars), and ECP values (Tables) for a) Alg1 calibration b) Alg2 calibration.	105
Figure 4.8: Experimental glycoform values (black dots), model prediction ranges (blue bars), and ECP values (Tables) after <i>MGat1</i> disruption.	107
Figure 4.9: Experimental glycoform values (black dots), model prediction ranges (blue bars), and ECP values (Tables) for baseline IgG glycoform.	109
Figure 4.10: Experimental glycoform values (black dots), model prediction ranges (blue bars), and ECP values (Tables) for CHO cells with <i>FUT8</i> knockout.	110
Figure 4.11: Experimental glycoform values (black dots), model prediction ranges (blue bars), and ECP values (Tables) for baseline IgG glycoform.	113
Figure 4.12: Experimental glycoform values (black dots), model prediction ranges (blue bars), and ECP values (Tables) for glycoform after <i>ST6Gal1</i> overexpression using ECPs calculated with (A) Alg1 and (B) Alg2. ...	115
Figure 5.1: Metabolic reaction network for co-substrate production.	123
Figure 5.2: Metabolic reaction network for co-substrate production with list of all ECPs (Table) and the corresponding reactions (grey arrows). For ECPs corresponding to bi-directional reactions, the direction of the grey arrow corresponds to the positive direction.	125

Figure 5.3: Workflow for calculating ECP values from a given glycoform distribution in Metabolism Analysis Mode.	127
Figure 5.4: Input metabolites (bars) required to match the glycoform measured after culture supplementation of galactose, ManNAc, and glucosamine (x-axis).	130
Figure 5.5: Flux of metabolites to CMP-NeuAc generation.	132
Figure 5.6: Required galactose flux into model as a function of UMG feeding level (units are in fold change of initial bolus concentration: 1 mM uridine, 5 mM galactose, and 0.002 mM MnCl ₂).	134
Figure 5.7: Percentage of material used to produce GDP-Fucose as a function of UMG feeding level (units are in fold change of initial bolus concentration: 1 mM uridine, 5 mM galactose, and 0.002 mM MnCl ₂).	136
Figure 5.8: Normalized molecules of glucose (left axis, blue) and galactose (right axis, orange) required by our model to produce each reported glycoform.	139
Figure 5.9: Glucose uptake rate required by our model to reproduce the glycoform measured for each experimental condition.	142
Figure 5.10: Specific glucose uptake rate of experimental CHO cells for 0, 4, and 8 mM Gln over a five-day batch culture (adapted from Burleigh et al. ...	143
Figure 5.11: Metabolic reaction network for co-substrate production marked with reactions used to calculate the ECPs ‘Man to Fuc’ (red), ‘Glc to Gal’ (yellow), ‘Glc Main’ (blue) and ‘CMP-NeuAc’ (purple).....	145
Figure 5.12: Distribution of ECP values calculated from glycoform measured on days 1, 3, and 5 of culture supplemented with 0, 4, and 8 mM Gln.....	146
Figure 6.1: GUI panel with labelled sections: 1) Mode of analysis, 2) Algorithm selection and setup, 3) Glycan search tool, and 4) Data input and results table.....	152
Figure 6.2: GUI when Metabolism Analysis Mode is selected.....	154
Figure B.1: Reactions corresponding to all ECP values.....	180

ABSTRACT

Advancements in biopharmaceutical research, development, and manufacturing have led to improved treatments for ailments including leukemia, multiple sclerosis, and diabetes. Over 70% of current therapeutics, composed of biopharmaceuticals such as recombinant proteins, are produced by Chinese Hamster Ovary (CHO) cells. One of the most important protein modifications enabled by CHO cells is glycosylation, or the sequential linkage of oligosaccharide chains of varying saccharidic composition. As glycosylation is a primary factor for determining biopharmaceutical safety and efficacy, there is a need for increased understanding and control of CHO cell glycosylation. *In silico* kinetic modeling has been shown to provide valuable insights that have furthered our understanding of glycosylation. However, due to the complexity of glycosylation, previous models have been built using a large number of parameters – many of which have unknown or assumed values. The large parametric demands of current glycosylation models make them difficult to use for *a priori* analysis and prediction of glycosylation.

The goal of this work is to develop a model that can be used to predict CHO cell glycosylation without the need for a large number of parameters. Successful development of this system will give users a way to study and predict glycosylation without requiring parameters for culturing conditions, therapeutic protein type, and CHO cell line modifications. This thesis establishes the framework for a novel method of modeling glycosylation in CHO cells through application of constraint-based analysis. Constraint-based analysis has been used as a low-parameter alternative to kinetic models for complex biological systems. This work creates a constraint-based model for glycosylation and introduces a series of discrete parameters that control

metabolite flux through the network. The system is used to predict changes in glycosylation due to cell line engineering and media optimization. Our model was found to successfully predict the effects of an *Mgat1* knockout on glycosylation patterns and validate the quantitative effects of *ST6Gal1* overexpression. Additionally, our model was able to validate the presence of media supplementations such as glucosamine, N-acetylmannosamine, galactose, uridine, and glutamine using only the initial and final glycosylation pattern (glycoform) as an input.

Unlike kinetic models, which are computationally demanding, complex, and require the *a priori* knowledge of >100 parameters, our model can make successful predictions and generate testable hypotheses for glycosylation using <20 parameters. We have designed a simple graphical user interface (GUI) which employs custom-built algorithms that can calibrate these parameters to any glycoform or simulate a glycoform given a set of parameters entered by the user. This work shows how development of a tunable constraint-based model can be used to predict and understand complex biological phenomena such as glycosylation. Finally, our model can generate testable hypotheses to explain the appearance of unexpected glycosylation patterns, resulting in an efficient way to control CHO cell glycosylation.

Chapter 1

INTRODUCTION

1.1 The Importance of Chinese Hamster Ovary Cells in Biopharmaceutical Development

Defined as products generated using modern molecular biological methods, biopharmaceuticals have experienced a three-fold growth in product approvals over the past two decades. The biopharmaceutical industry, with only 16 product approvals from 1990 to 1994, exhibited 50 product approvals from 2010-2014 and annual sales surpassing \$100 billion (Walsh, 2014). Over 70% of current biopharmaceutical products are manufactured in CHO cells, making them the predominate host cell line used in the biopharmaceutical industry (Kim et al., 2012).

The use of CHO cells to manufacture biopharmaceuticals offers increased standardization, safety, and biopharmaceutical efficacy, which contribute to its widespread use in industry. CHO cells have been adapted for growth in both large-scale suspension culture and serum-free media (Lai et al., 2013). This allows CHO cells to produce biopharmaceuticals with greater reproducibility. Additionally, human-specific viruses are unable to replicate in CHO cells, dramatically increasing the safety of drugs produced in CHO cells compared to other platforms (Boeger et al., 2005). CHO cells have also been subjected to decades of research towards the development of highly optimized gene amplification systems, which allow CHO cells to exhibit titers exceeding 10 g/L (Kim et al., 2012). These systems involve simultaneous

expression of a gene of interest and an enzyme required for optimal CHO cell growth, such as dihydrofolate reductase or glutamine synthase (Cacciatore et al., 2010), and have been used to generate stable, high-producing CHO cell lines within 6-12 months (Lai et al., 2013). One of the most important advantages of CHO cells is their ability to produce proteins with human-compatible post-translational modifications, such as glycosylation (Ghaderi et al., 2012). As glycosylation affects the cytotoxicity, immunogenicity, half-life, and efficacy of many biopharmaceuticals, there is a strong need to understand CHO cell glycosylation to maximize biopharmaceutical safety and efficacy.

1.2 Glycosylation Alters Protein Therapeutic Performance

Glycosylation, or the process by which oligosaccharide molecules (glycans) bind to specific sites on a protein backbone, is used by cells to modify the structure and function of over 50% of human proteins (Walsh & Jefferis, 2006). For monoclonal antibodies (mAbs), these glycans have a dramatic impact on the biopharmaceutical's effector function(s): actions taken by the patient's immune system when it recognizes the mAb *in vivo* (Jiang et al., 2011). Examples of important effector functions include antibody-dependent cell-mediated cytotoxicity (ADCC) and complement-dependent cytotoxicity (CDC), both of which have been found to be affected by the presence of specific glycans (Hossler et al., 2009). These effector functions use different immunogenic pathways to destroy the biopharmaceutical's target, and different biopharmaceuticals require different intensities of effector functions for successful treatment (Chan & Carter, 2010).

Glycosylation also affects the clearance rate and activity of non-mAb biopharmaceuticals. Human erythropoietin (EPO) is a glycosylated cytokine protein manufactured to treat anemia (Walsh, 2014). The presence of sialic acid residues on terminal branches of EPO were found to increase stability and activity of the protein (Takeuchi et al., 1988). Additionally, sialic acid residues have been shown to increase the half-life of a therapeutic protein from <2 min to >5 hrs in EPO-producing CHO cells (Erbayraktar et al., 2003; Kompella et al., 1991). As glycosylation affects the safety and efficacy of both mAb and non-mAb therapeutics, it is important to understand the underlying mechanisms of glycosylation and how they can be controlled through biopharmaceutical manufacturing.

The link between biopharmaceutical development techniques and glycosylation patterns has driven scientists to search for different strategies to optimize the glycoform of CHO cells for various biopharmaceuticals. For example, basic, unprocessed glycans are desired for therapeutics that interact with the mannose-mediated uptake pathways of cells *in vivo* (Betting et al., 2009; Lam et al., 2005). As the formation and maturation of glycans is dictated by many independently-acting enzymes (glycosyltransferases) rather than DNA transcription and translation, the process is intrinsically less regulated and controllable than protein synthesis. For this reason, there has been a significant amount of work done to further our understanding of key factors that control glycosylation in CHO cells. For example, Umaña et al. developed a tetracycline-regulated expression system in CHO cells to overexpress the glycosyltransferases N-acetylglucosaminyltransferase III and V to increase the *in vivo* activity of certain biopharmaceuticals. Although their experiment resulted in high levels of both enzymes and an increased glycan antennarity, severe growth inhibition

was observed in cultures with high expression of the glycosyltransferases (Umaña et al., 1999). Weikert et al. overexpressed galactosyltransferase and sialyltransferase in two different CHO cell lines, each producing a different type of therapeutic glycoprotein in an effort to increase the proteins' *in vivo* half-life. They found that the experiment increased glycan sialylation to over 90%, resulting in drugs with a significantly longer half-life (Weikert et al., 1999). Additionally, no growth inhibition was observed, suggesting that the negative effects of glycosyltransferase overexpression on CHO cell growth is highly variable and dependent on both the cell line and the extent of overexpression. Overall, attempts to experimentally improve a glycoform have been met with mixed results, and quantitatively controlling the glycoform without a thorough knowledge of all kinetic and metabolic factors that influence glycosylation has proved difficult (Sha et al., 2016).

1.3 Mathematical Models Can Improve Glycosylation Engineering

Creating mathematical models is a useful way to gain a mechanistic understanding of glycosylation. Glycosylation is a complex process that takes place over multiple length and time scales, spans across various cellular organelles, and is regulated by many different proteins (Galleguillos et al., 2017). Despite this, glycosylation modeling has improved greatly over time, with the most recent models able to describe not only the kinetics of glycan maturation in the Golgi, but also the effect of variations in protein production and metabolism to predict glycosylation patterns with >95% accuracy (Jiminez del Val et al., 2016).

Despite the advancements made in CHO cell glycosylation modeling, even the most advanced models cannot make accurate quantitative predictions *a priori* thanks to the large number of kinetic and biological parameters needed to describe glycosylation. The majority of glycosylation models are based on kinetic descriptions of both glycosyltransferase action on a glycoprotein as well as the generation of all glycosylation-related sugar molecules (Umaña et al., 1997; Krambeck et al., 2005). Many model parameters cannot be accurately measured, and as a result the majority of these parameters have unknown or assumed values. Due to the complexity of estimating accurate kinetic parameters in glycosylation models, their ability to predict CHO cell glycosylation without detailed quantification of the enzymatic and metabolic state of the CHO cell line is limited.

Recently, a novel glycosylation modeling tool has been developed in the Lee lab (Kremkow & Lee, submitted) that uses simplified kinetics and a high-level genomic network to predict glycosylation patterns using a significantly smaller set of parameters. While the current kinetic models for glycosylation requires over 100 parameters to model glycosyltransferase activity (Krambeck et al., 2005), the Glyco-Mapper uses only 37. This low-parameter strategy allows it to generate accurate predictions *a priori* using only the glycoform as an input; this has not been accomplished by any kinetic models to our knowledge. Though this model (named the ‘Glyco-Mapper’) makes accurate predictions with appropriate parameter adjustments, it is currently not sophisticated enough to generate quantitative glycoform predictions. As this model can only predict which glycans are present or absent, it cannot calculate the abundance of each glycan. Development of an improved Glyco-Mapper that can

implement quantitative analysis of glycosylation using a low number of parameters would enhance our ability to efficiently predict and study glycosylation in CHO cells.

1.4 Project Goals

The goal of our work is to develop a quantitative model for glycosylation that can successfully predict the effect(s) of changes in glycosyltransferase expression and media composition using a low number of parameters (<40). Specifically, we would like to implement a comprehensive set of algorithms for quantitative analysis, establish an appropriate reaction network, and develop a graphical user interface (GUI) that would result in an easy-to-use and quantitatively accurate model for glycosylation analysis and prediction. There are three specific objectives in this work:

- (1) Develop a framework with which to quantitatively analyze and predict CHO cell N-glycosylation patterns as a function of glycosyltransferase concentration and activity. Validate this model by predicting various glycosyltransferase gene knockout and overexpression experiments in literature.
- (2) Apply the framework developed in Goal 1 to model glycosylation patterns in CHO cells as a function of sugar co-substrate availability. Validate by using our model to study the metabolic requirements of various media supplementation experiments in literature.
- (3) Develop a user-friendly GUI for the model, which connects the Matlab scripts and Excel files used to build our model together for easy operation and modification.

1.5 Scope of Work

This thesis allows the reader to gain a thorough understanding of our model as well as its validation process. Chapter 2 explains the necessary background information for the subjects addressed in this work: glycosylation biology, glycosylation modeling, and constraint-based analysis. Chapter 3 outlines the organizational structure of our model and the establishment of our parameter-driven control system. Chapters 4 and 5 are detailed explanations of the model's main two functions, which analyze glycosyltransferase activity and co-substrate generation, respectively. Chapter 6 describes the creation and implementation of a user-friendly GUI and all associated functions. Finally, Chapter 7 addresses model shortcomings and offers possible solutions for future research.

REFERENCES

Betting, D. J., Mu, X. Y., Kafi, K., McDonnell, D., Rosas, F., Gold, D. P., & Timmerman, J. M. (2009). Enhanced immune stimulation by a therapeutic lymphoma tumor antigen vaccine produced in insect cells involves mannose receptor targeting to antigen presenting cells. *Vaccine*, 27(2), 250-259.

Boeger, H., Bushnell, D. A., Davis, R., Griesenbeck, J., Lorch, Y., Strattan, J. S., ... & Kornberg, R. D. (2005). Structural basis of eukaryotic gene transcription. *FEBS letters*, 579(4), 899-903.

Cacciatore, J. J., Chasin, L. A., & Leonard, E. F. (2010). Gene amplification and vector engineering to achieve rapid and high-level therapeutic protein production using the Dhfr-based CHO cell selection system. *Biotechnology advances*, 28(6), 673-681.

Chan, A. C., & Carter, P. J. (2010). Therapeutic antibodies for autoimmunity and inflammation. *Nature Reviews Immunology*, 10(5), 301.

603.

Erbayraktar, S., Grasso, G., Sfacteria, A., Xie, Q. W., Coleman, T., Kreilgaard, M., ... & Yilmaz, O. (2003). Asialoerythropoietin is a nonerythropoietic cytokine with broad neuroprotective activity in vivo. *Proceedings of the National Academy of Sciences*, 100(11), 6741-6746.

Galleguillos, S. N., Ruckerbauer, D., Gerstl, M. P., Borth, N., Hanscho, M., & Zanghellini, J. (2017). What can mathematical modelling say about CHO metabolism and protein glycosylation?. *Computational and structural biotechnology journal*, 15, 212-221.

Ghaderi, D., Zhang, M., Hurtado-Ziola, N., & Varki, A. (2012). Production platforms for biotherapeutic glycoproteins. Occurrence, impact, and challenges of non-human sialylation. *Biotechnology and Genetic Engineering Reviews*, 28(1), 147-176.

Hossler, P., Khattak, S. F., & Li, Z. J. (2009). Optimal and consistent protein glycosylation in mammalian cell culture. *Glycobiology*, 19(9), 936-949.

Jiang, X. R., Song, A., Bergelson, S., Arroll, T., Parekh, B., May, K., ... & Schenerman, M. (2011). Advances in the assessment and control of the effector functions of therapeutic antibodies. *Nature reviews Drug discovery*, 10(2), 101.

Jimenez del Val, I., Nagy, J. M., & Kontoravdi, C. (2011). A dynamic mathematical model for monoclonal antibody N-linked glycosylation and nucleotide sugar donor transport within a maturing Golgi apparatus. *Biotechnology progress*, 27(6), 1730-1743.

Kim, J. Y., Kim, Y. G., & Lee, G. M. (2012). CHO cells in biotechnology for production of recombinant proteins: current state and further potential. *Applied microbiology and biotechnology*, 93(3), 917-930.

Kompella, U. B., & Lee, V. H. (1991). Pharmacokinetics of peptide and protein drugs. In *Peptide and protein drug delivery*(Vol. 4, pp. 391-484). Marcel Dekker New York.

Krambeck, F. J., & Betenbaugh, M. J. (2005). A mathematical model of N-linked glycosylation. *Biotechnology and Bioengineering*, 92(6), 711-728.

Kremkow, B., Lee, K. (2017) (Submitted)

Lai, T., Yang, Y., & Ng, S. K. (2013). Advances in mammalian cell line development technologies for recombinant protein production. *Pharmaceuticals*, 6(5), 579-

Lam, J. S., Mansour, M. K., Specht, C. A., & Levitz, S. M. (2005). A model vaccine exploiting fungal mannosylation to increase antigen immunogenicity. *The Journal of Immunology*, 175(11), 7496-7503.

Sha, S., Agarabi, C., Brorson, K., Lee, D. Y., & Yoon, S. (2016). N-Glycosylation design and control of therapeutic monoclonal antibodies. *Trends in biotechnology*, 34(10), 835-846.

Takeuchi, M., Takasaki, S., Miyazaki, H., Kato, T., Hoshi, S., Kochibe, N., & Kobata, A. (1988). Comparative study of the asparagine-linked sugar chains of human erythropoietins purified from urine and the culture medium of recombinant Chinese hamster ovary cells. *Journal of Biological Chemistry*, 263(8), 3657-3663.

Walsh, G. (2014). Biopharmaceutical benchmarks 2014. *Nature biotechnology*, 32(10), 992.

Walsh, G., & Jefferis, R. (2006). Post-translational modifications in the context of therapeutic proteins. *Nature biotechnology*, 24(10), 1241.

Weikert, S., Papac, D., Briggs, J., Cowfer, D., Tom, S., Gawlitzek, M., ... & Eppler, S. (1999). Engineering Chinese hamster ovary cells to maximize sialic acid content of recombinant glycoproteins. *Nature biotechnology*, 17(11), 1116.

Umaña, P., & Bailey, J. E. (1997). A mathematical model of N-linked glycoform biosynthesis. *Biotechnology and bioengineering*, 55(6), 890-908.

Chapter 2

BACKGROUND OF GLYCOBIOLOGY AND MODELING TECHNIQUES

2.1 Introduction

This thesis covers a variety of different biological and computational subjects and requires sufficient knowledge in three areas: the process of N-glycosylation, past and current *in silico* glycosylation modeling techniques, and constraint-based analysis. This chapter begins with a walkthrough of N-glycan synthesis, formation, and maturation. Next, a comprehensive literature review is included to explain glycosylation modeling from the first models to the current state of the art. Finally, this chapter concludes with a thorough introduction and explanation of constraint-based analysis techniques in the context of metabolic network modeling.

2.2 The Process of N-Glycosylation

N-glycosylation takes place during protein synthesis in the endoplasmic reticulum (E.R.) and continues until the mature glycan leaves the Golgi Apparatus upon protein secretion. This process can be divided into three sections. The first includes the initial linkage of the starting glycan to the appropriate amino acid sequence on a developing glycoprotein and subsequent transportation through the E.R. The second section describes glycoprotein transport through the Golgi and glycan modifications via a series of Golgi-bound enzymes. The last section involves the synthesis and transport of each saccharide building block from the cytosol to the Golgi Apparatus.

2.2.1 Glycan Synthesis and E.R. Transport

When a protein is being synthesized, the appropriate strand of RNA is taken in and scanned by ribosomes in the rough E.R. Each ribosome uses the appropriate amino acids to convert its RNA into protein. This newly synthesized polypeptide has a specific sequence that binds to an available signal recognition particle (SRP). This SRP binds both the ribosome and its protein to the plasma membrane of the rough E.R. Once bound to the plasma membrane, a translocation channel allows the new protein to be funneled into the E.R. lumen. If this process is performed correctly and there are no errors in protein synthesis, glycosylation can begin immediately (Brooks et al, 2002).

Glycosylation begins with the synthesis of an unprocessed glycan intermediate. The structure of this glycan (with the chemical formula $\text{Glc}_3\text{Man}_9\text{GlcNAc}_2\text{-P-P-Dol}$) consists of a multi-branched, high mannose oligosaccharide bound to a dolichol phosphate complex (*-P-P-Dol). This complex is bound to the E.R. membrane during synthesis and is shown in Figure 2.1. Synthesis begins with the action of N-acetylglucosaminylphosphotransferase to link a GlcNAc molecule to the phosphodolichol complex. Next, N-acetylglucosaminyltransferase links another GlcNAc to the first in a $\beta(1, 4)$ attachment, forming the $\text{GlcNAc-GlcNAc-P-P-Dol}$ complex. The mannosyltransferase I-V enzymes are next to act, linking five mannose sugar molecules to the glycan in the pattern shown in Figure 2.1. The entire oligosaccharide is then “flipped” to the other side of the E.R. in a process that is still poorly understood. This mechanism is, however, necessary for the next step in the glycan’s maturation. After the flip, mannosyltransferases and glucosyltransferases

complete the formation of the $\text{Glc}_3\text{Man}_9\text{GlcNAc}_2\text{-P-P-Dol}$ glycan intermediate. The glycan can now be attached to its target polypeptide chain.

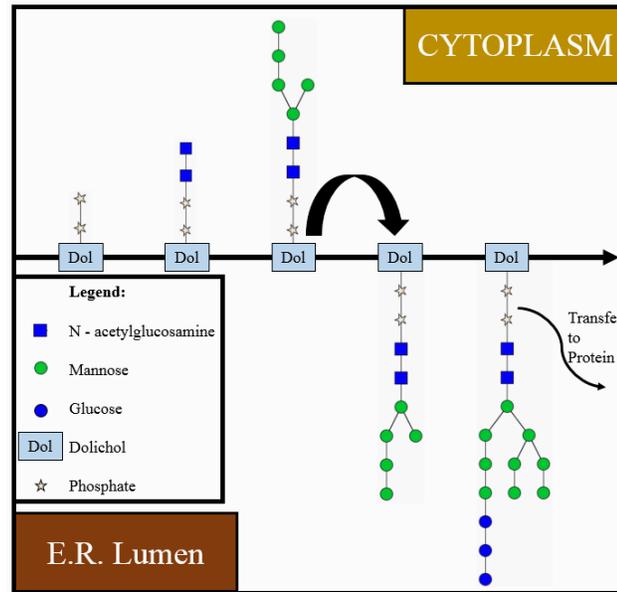


Figure 2.1: Formation of $\text{Glc}_3\text{Man}_9\text{GlcNAc}_2\text{-P-P-Dol}$ in the E.R.

N-linkage of a glycan to a polypeptide is a highly specific process. To become glycosylated, the growing protein must have an appropriate consensus sequence of peptides along its backbone. For N-glycosylation, glycans can only be linked to an asparagine (Asn) residue in the sequence $\text{Asn-X}_n\text{-Serine/Threonine}$, where X_n represents any number of amino acids that are *not* proline. Another type of glycosylation (O-glycosylation), involves a glycan linking to a Serine or Threonine residue. However, O-glycosylation is not within the scope of this thesis. The enzyme that is instrumental in this transfer is called oligosaccharyltransferase (OST). Once the

intermediate glycan is fully formed, an OST removes the glycan by breaking the link between the first GlcNAc residue and P-P-Dol. The P-P-Dol residue remains linked to the lumen of the E.R. to process other glycans, and the oligosaccharide is then linked to the protein via OST. The final stage of glycan processing in the E.R. involves the sequential trimming of all glucose molecules via glucosidase to create a $\text{Man}_9\text{GlcNAc}_2$ motif linked to the glycoprotein's Asn residue. The glycoprotein is then shuttled to the Golgi for further modification.

2.2.2 Intra-Golgi Protein Transport

Before explaining the next stage of glycosylation, it is imperative to first gain an understanding of protein transport through the Golgi. The Golgi consists of a series of flat, disk-like compartments called cisternae. Blocks of cisternae are categorized into four groups, denoted the *cis*-Golgi, *medial*-Golgi, *trans*-Golgi, and the Trans-Golgi Network (TGN). The *cis*-Golgi is closest to the E.R., while the TGN is the farthest. Each Golgi stack is populated with a unique composition of enzymes including glycosidases and glycosyltransferases (which respectively cleave and attach saccharide molecules from/to the glycan), and enzymes are shuttled to and from compartments through transport vesicles. Proteins are transported from the *cis*-Golgi to the TGN, however the methodology of this transfer has been hotly debated. Though there is no dispute regarding the role that coatamer (COPI) transport vesicles play in retrograde protein transport (*trans* to *cis* direction), there is evidence for multiple unique theories on how proteins are transported in an anterograde (*cis* to *trans*) direction (Pelham et al., 2000). There are two paradigms that explain protein transport

through the Golgi: vesicular transport theory and cisternal maturation theory (Figure 2.2).

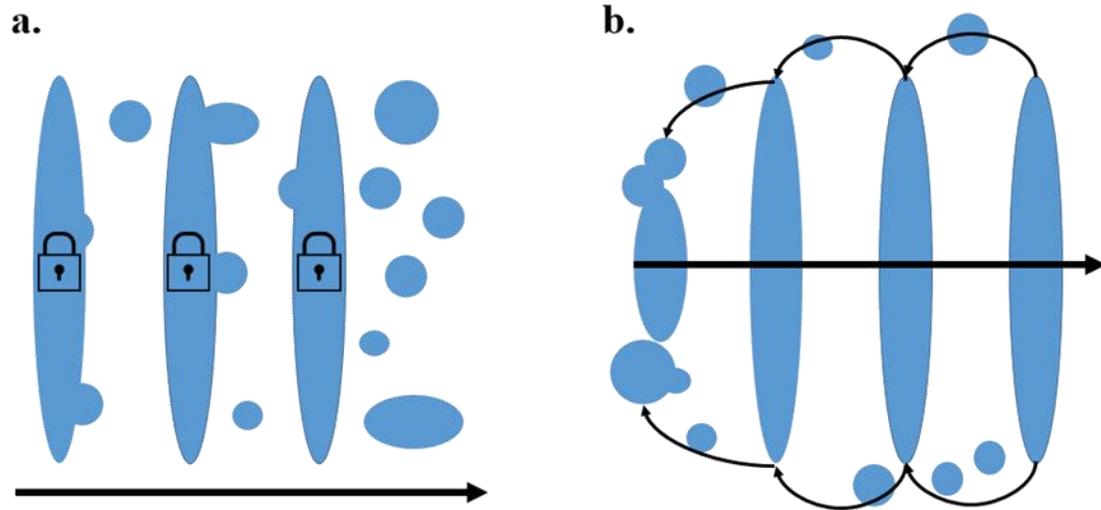


Figure 2.2: Visual representation of protein transport via (A) vesicular transport or (B) cisternal maturation.

Vesicular transport theory was one of the first attempts to describe anterograde protein transport (Farquhar, 1985). This theory postulates that each Golgi stack is stationary and populated with a relatively fixed number of enzymes. Proteins travel from the *cis-Golgi* to the *trans-Golgi* through COPI transport vesicles budding off of and into each cisternae. Evidence fits this theory nicely, as a large fraction of COPI vesicles were observed to contain a protein that is consistently transported to the plasma membrane, making its transport anterograde by extension (Orci et al., 1986). Transport of proteins across vesicles has also been observed, adding further support to the vesicular transport model (Balch et al., 1984). However, lack of a clear

anterograde transport mechanism for COPI vesicles made the theory incomplete. Additionally, this theory was unable to provide an explanation for the transport of cargo that is larger than COPI vesicles (Bonfanti et al., 1998).

The second theory, cisternal maturation, asserts that rather than the protein moving through stationary cisternae, the protein is stationary and instead all cisternae are moving through the Golgi in a *cis* to *trans* direction. In a conveyor-belt fashion, the protein is carried through the Golgi by the independent movement of a single cisternae. COPI transport vesicles then dynamically change the enzyme content of each stack in a retrograde fashion as the stacks move through the Golgi (Glick et al., 1997). This theory is consistent with the established behavior of COPI vesicles and offers an explanation as to how oversize cargo progresses through the Golgi stack. However, the rate of transport via cisternal maturation in pulse-chase experiments are drastically slower than for other reported proteins, suggesting this cannot be the sole mechanism of protein transport (Bonfanti et al., 1998).

Over the years, glycosylation models have reflected this polarization in the paradigms for protein transport through the Golgi. Originally, glycosylation models represented the Golgi as a series of continuous-stirred tank reactors (CSTRs), which mimic vesicular transport in that material is transferred between stationary stacks in an anterograde fashion (Umaña & Bailey, 1997; Krambeck & Betenbaugh, 2005). Over time, glycosylation models began adopting a cisternal maturation approach, representing the Golgi as a plug-flow reactor (PFR) in which incoming glycoproteins progress through a changing glycosyltransferase distribution in the same stack (i.e. segment of the PFR). As both representations give comparable results when applied to

predict glycosylation patterns, it is likely that the true mechanism(s) of protein transport through the Golgi is a combination of both theories (Jimenez et al., 2011).

2.2.3 Glycan Modification through Glycosyltransferase Processing

Regardless of the true nature of protein transport through the Golgi stacks, the enzymes responsible for processing any incoming N-glycan are well documented. A non-processed and fully-processed N-glycan are shown in Figure 2.3. The glycan structure consists of two GlcNAc residues linked sequentially ($\beta(1, 4)$ linkage) to an Asparagine residue on the protein backbone. The second GlcNAc residue is attached to a mannose residue, also through a $\beta(1, 4)$ linkage. This mannose is bound to two separate mannose residues in an $\alpha(1, 3)$ and $\alpha(1, 6)$ linkage, respectively. This tri-mannose structure makes up the core mannose section of a glycan. Before processing, six additional mannose residues are linked in a combination of $\alpha(1, 3)$ and $\alpha(1, 6)$ linkages to create what is known as the M9 glycan (Figure 2.3). A fully-processed N-glycan has all mannose residues removed except for the three core residues. Additionally, a maximum of four GlcNAc residues are bound in a combination of $\beta(1, 2)$ and $\beta(1, 4)$ linkages. Bound to the GlcNAc residues are up to four galactose residues, and sialic acid residues are the final, terminal sugars to be found on an N-glycan (Figure 2.3). These sugars are removed from and added to the glycan through the action of various independent enzymes, and in many cases a large portion of the glycoform is incompletely processed due to the kinetic limitations of enzyme processing.

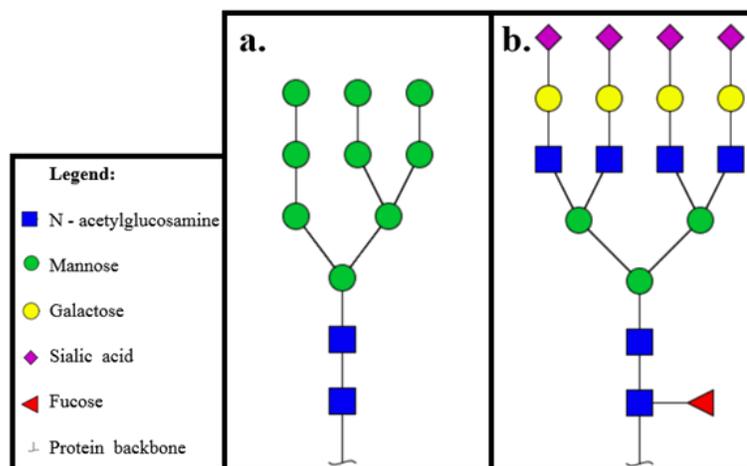


Figure 2.3: (A) Visual representation of an unprocessed N- glycan upon exiting the E.R. (B) Visual representation of a terminally-processed N-glycan exiting the TGN.

There are ten major enzymes (glycosidases and glycosyltransferases) involved in glycosylation. The names of these enzymes are mannosidase I and II (ManI and ManII), N-acetylglucosaminyltransferase I through V (GnTI-GnTV), fucosyltransferase (FucT), galactosyltransferase (GalT), and sialyltransferase (SialT). In this report, we will refer to all enzymes that change the chemical structure of a glycan as glycosyltransferases for convenience. These glycosyltransferases are membrane-bound and distributed throughout the Golgi in roughly compartmental locations (Rabouille et al., 1995; Opat et al., 2001) (Figure 2.4). The cis-Golgi is composed primarily of ManI and GnTI, while the medial-Golgi contains GnTI-V and FucT. The trans-Golgi and TGN consist of GnTIV-V, FucT, GalT, and SialT (Jiminez et al. 2011; Rabouille et al., 1995). In addition to glycosyltransferase localization, the physical structure of the glycoprotein can limit the action of certain glycosyltransferases, preventing some glycoforms from being achieved. For example,

the physical structure of an IgG protein prohibits the action of GnTIII, GnTIV, and GnTV, resulting in solely biantennary glycans (Higel et al., 2016). Conversely, EPO, with a more exposed glycosylation site, can easily achieve tetraantennary glycoforms and has a high degree of sialylation (Yang & Butler, 2002).

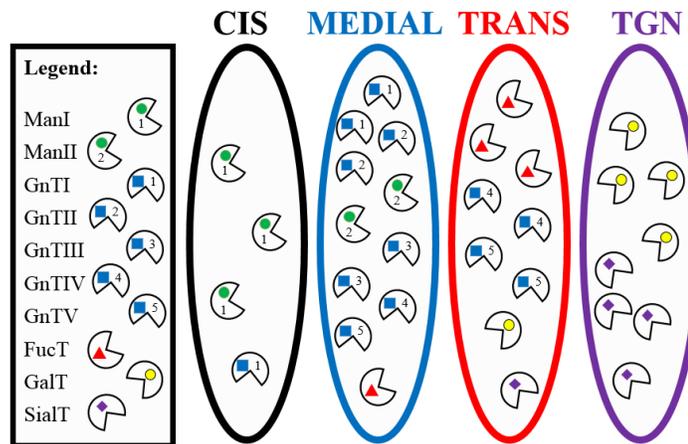


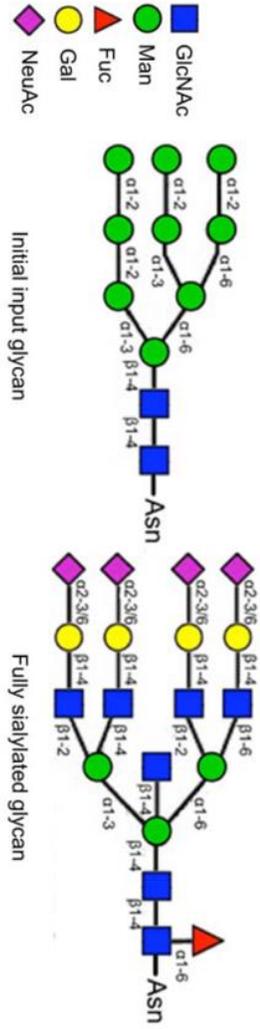
Figure 2.4: Approximate localization of major glycosyltransferases for N-glycosylation in the Golgi.

Another key property of glycosyltransferases is that many of them can only act on specific terminal saccharide residues (substrates) on the glycan. For example, GnTI can only add a GlcNAc residue to the $\alpha(1, 3)$ branch of a terminal mannose residue. While GnTIV can only add a GlcNAc saccharide to the $\alpha(1, 6)$ -linked mannose branch. All major N-glycosylation enzymes as well as their function and substrate requirements are shown in Table 2.1. Substrate specificity combined with glycosyltransferase localization culminates in a reaction network of reduced complexity. However, the combinatorial explosion that occurs from the action of key glycosyltransferases results in a maximum network of thousands of potential glycans

and hundreds of thousands of reactions. Despite this complexity, current research has mapped a rough outline of a glycan's path through the Golgi as well as the conditions under which the network exhibits a combinatorial expansion and collapse.

Table 2.1: Diagram of initial and final glycan as well as major glycosyltransferase function and substrate requirements (glycan images: Kim et al., 2009).

Enzyme	ManI	GnTI	ManII	GnTII	GnTIV	GnTV	FucT	GnTIII	GaII	StnT
Function	Trims $\alpha(1,2)$ -linked mannose residue	Adds $\beta(1,2)$ -linked GlcNAc residue	Trims $\alpha(1,3/6)$ -linked mannose residue	Adds $\beta(1,2)$ -linked GlcNAc residue	Adds $\beta(1,4)$ -linked GlcNAc residue	Adds $\beta(1,6)$ -linked GlcNAc residue	Adds $\alpha(1,6)$ -linked fucose residue	Adds bisecting GlcNAc residue	Adds $\beta(1,4)$ -linked galactose residue	Adds $\alpha(2,3/6)$ -linked NeuAc residue
Substrate and restrictions	- Free $\alpha(1,2)$ -linked mannose residue	- M5 glycan only	- Free $\alpha(1,3/6)$ -linked mannose residue - Glycan previously acted on by GnTI, free residue	- Free $\alpha(1,6)$ -linked mannose residue - Glycan previously acted on by GnTI, free residue	- No $\beta(1,4)$ -linked GlcNAc residue linked to $\alpha(1,3)$ mannose	- No $\beta(1,6)$ -linked GlcNAc residue linked to $\alpha(1,6)$ mannose	- No $\alpha(1,6)$ -linked fucose residue - Glycan previously acted on by GnTI	- bisecting GlcNAc residue - Glycan previously acted on by GnTI	- Free GlcNAc residue	- Free galactose residue



The first modification undergone by an incoming $\text{Man}_9\text{GlcNAc}_2\text{-Asn}$ (M9) glycan is the sequential trimming of up to five $\alpha(1, 2)$ -linked mannose residues by mannosidase-I (ManI). ManI is a key glycosyltransferase, because the ideal product glycan ($\text{Man}_5\text{GlcNAc}_2\text{-Asn}$) is necessary for all future modifications. The reason for this is that GnTI can only bind to an $\alpha(1, 3)$ -linked mannose residue if all other $\alpha(1, 2)$ -linked mannose residues are removed. If the glycan is not processed completely by ManI, it is likely that it will remain in its final form (described as high mannose type glycan) for the duration of the protein's trip through the Golgi. Providing it is completely processed, GnTI can successfully add a $\beta(1, 4)$ -linked GlcNAc residue to the glycan and further modifications can ensue.

After the creation of $\text{GlcNAc}_1\text{Man}_5\text{GlcNAc}_2\text{-Asn}$, a second mannosidase (ManII) can act. This glycosidase cleaves all terminal $\alpha(1, 3)$ and $\alpha(1, 6)$ -linked mannose residues. Complete processing of a glycan by this enzyme results in a $\text{GlcNAc}_1\text{Man}_3\text{GlcNAc}_2\text{-Asn}$, and all glycans that are built from this base are called complex type glycans. However, if ManII does not remove all eligible mannose residues, a $\text{GlcNAc}_1\text{Man}_4\text{GlcNAc}_2\text{-Asn}$ glycan is created, which are called hybrid type glycans. After this point, the combinatorial explosion of the reaction network begins. Four additional enzymes can add another GlcNAc residue to terminal mannose branches (GnTII, GnTIV, and GnTV) or the core mannose (GnTIII) of the glycan. The terminal GlcNAc residues denote the glycan's antennarity. For example, a glycan with two GlcNAc residues linked to a terminal mannose is called biantennary glycan, and if there are three GlcNAcs it is called a triantennary glycan. All N-glycans have a maximum of four possible terminal antennae, and this terminology is used to describe

glycoforms as research has shown the antennarity of a glycan to affect critical therapeutic properties *in vivo*.

There are many other enzymes that have the capability to modify the glycoform other than mannosidases and N-acetylglucosaminyltransferases. Some major glycosyltransferase families are fucosyltransferase, galactosyltransferase. And sialyltransferase. The enzymes respectively add fucose, galactose, and sialic acid residues to the glycan in a manner compatible with each enzyme's substrate specifications (Table 2.1). All these enzymes work together to create thousands of potential glycans and the large heterogeneity associated with CHO cell glycosylation.

2.2.4 Metabolic Production of Sugar Co-substrates for Glycosylation

Media supplementation to control glycosylation is a growing field of study (McAtee et al., 2014), as it would be a cheaper and more flexible method of control than cellular engineering. This form of glycosylation control aims to affect the distribution of intracellular co-substrates for glycosylation rather than the activity/concentration of glycosyltransferases. Each glycosyltransferase requires the correct saccharide molecule (co-substrate) to successfully alter the chemical composition of a compatible glycan substrate. These co-substrates are simple saccharides such as fucose, galactose, N-acetylglucosamine, and N-acetylneuraminic (sialic) acid; they are produced from a small set of reactions in the central carbon metabolism (CCM) of CHO cells.

The CCM is a series of chemical processes within the cell that is primarily responsible for ATP, sugar, and cofactor generation. The three major pathways that

compose the CCM are known as glycolysis, the pentose phosphate pathway (PPP), and the tricarboxylic acid (TCA) cycle (Ahn & Antoniewicz, 2011). Essentially, glucose is converted step-wise into pyruvate in the glycolysis pathway. At the beginning of glycolysis, a portion of the intermediate glucose-6-phosphate is funneled into the PPP for protein production. The pyruvate formed at the end of glycolysis is one of the major inputs to the TCA cycle. The TCA cycle takes in pyruvate as well as glutamine to generate ATP. These three processes work together to produce all the necessary cellular materials for healthy growth.

The glycosylation pathway is a minor branch off of glycolysis. Though crucial for successful glycosylation, scientists cannot currently quantitatively analyze metabolic flux through this pathway due to both a lack of metabolic flux analysis techniques for glycosylation as well as its relatively small activity compared to glycolysis (Badur et al., 2015). The glycosylation pathway is responsible for converting glucose, fructose, galactose, and mannose into sufficient amounts of each co-substrate to ensure complete glycosylation. Figure 2.5 shows a map of our current understanding of the glycosylation metabolic network.

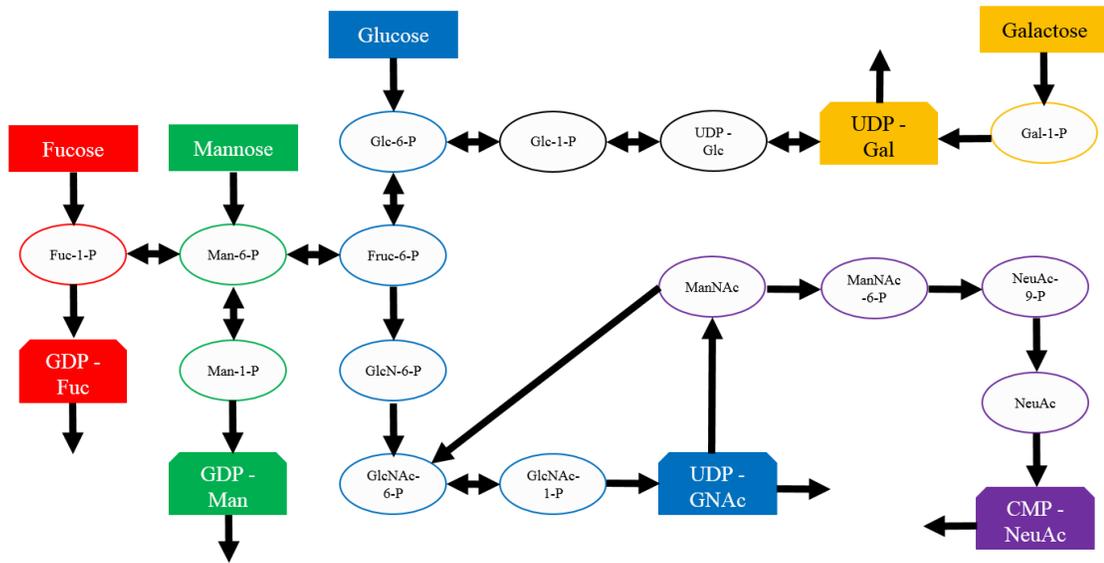


Figure 2.5: Metabolic network describing the conversion of simple sugars to co-substrates used in glycosylation (UDP-Gal, GDP-Fuc, GDP-Man, UDP-GNAc, CMP-NeuAc).

As shown in Figure 2.5, each co-substrate comprises the saccharide of interest (mannose, fucose, galactose, fructose, sialic acid) connected to a nucleotide transporter motif: guanosine diphosphate (GDP) for fructose and mannose, uridine diphosphate (UDP) for glucose and galactose, and Cytidine-5-monophosphate (CMP) for sialic acid. These molecules are shuttled into the Golgi by a series of transporter proteins and subsequently recognized by the correct glycosyltransferase. The glycosyltransferase then cleaves the nucleotide motif from the molecule and promotes binding of the sugar to its respective substrate. However, the time scale for co-substrate transport and reaction is often slower than the time scale for co-substrate generation, and in most cases co-substrates accumulate in the cytosol in what is referred to as sugar or co-substrate ‘pools’, each pool waiting to interact with its appropriate transporter protein.

Although this process seems straightforward, the required action of many independent metabolic enzymes, transporter proteins, and sugar pools make glycosylation metabolism difficult to predict. Increasing the size of a specific sugar pool through appropriate feeding of that sugar in the media is not always guaranteed to increase the rate of transfer for that sugar on incoming glycoproteins. For example, Yang et al. supplemented their CHO culture with glucosamine, a precursor to UDP-GlcNAc. Although supplementation increased the UDP-GlcNAc pool size, a decrease in glycoform antennarity was observed. As GlcNAc residues are responsible for increasing the antennarity of a glycan, this implies that the rate of transfer of GlcNAc on compatible glycans decreased with increased co-substrate material. Additionally, Gramer et al. performed a titrated feeding of uridine (Urd), manganese, and galactose to their CHO cells. They found that the galactosylation rate exhibited diminishing returns, possibly due to the inevitable enzymatic bottleneck of galactosyltransferase in the Golgi.

Despite these irregularities, the literature has isolated general trends that have served to guide glycosylation control in CHO cells. Media supplementation of galactose was found to increase the percentage of galactosylated and sialylated glycans, suggesting that the co-substrate UDP-Gal may have been the limiting factor for complete glycosylation processing in those cell lines (Wong et al., 2010). Addition of glucosamine and N-acetylmannosamine (ManNAc) was found to increase the intracellular UDP-GlcNAc pool and subsequently increase the number of GlcNAc residues in the glycoform (Wong et al., 2010). Similarly, addition of glucosamine has been found to increase the UDP-GlcNAc pool as well as increase glycosylation antennarity in previous studies (Gawlitzeck et al., 1998; Grammatikos et al., 1998;

Baker et al., 2001). These effects are largely due to the role of glucosamine in UDP-GlcNAc synthesis (McAtee et al., 2014). Attempts to control sialylation have been met with higher degrees of uncertainty. While ManNAc supplementation has been found to significantly elevate sialylation levels (Gu and Wang, 1998), glucosamine supplementation has been observed to have either little effect or a negative effect on the percentage of sialylated glycans (Grammatikos et al., 1998; Gawlitzek et al., 2000; Baker et al., 2001).

There has recently been work to understand and decouple the effects of feeding various sugar precursors and co-factors in CHO cells. For example, St. Amand et al. completed a two-factor statistical design of experiments (DOE) to understand the effect of various media additives in greater detail with respect to glycosylation. Manganese, galactose, and NH_4 were all supplemented individually and in combinations while the resulting glycoform for each culture was quantified. Manganese and galactose were found to increase the percentage of galactosylation in the culture, and NH_4 resulted in an overall decrease of complex glycans (St. Amand et al., 2014). Manganese has been previously shown to aid in the action of galactosyltransferase (Gramer et al., 2011), which is why it exhibits similar effects to supplementing galactose directly. NH_4 is a cell waste product that has been shown to inhibit sialylation as well as decrease antennary (Chen & Harcum, 2006; Grammatikos et al., 1998), consistent with St. Amand et al.'s findings. Despite these previous findings, the precise quantitative effects of media supplementation are difficult to predict. St. Amand et al. found that media supplementation resulted in modified glycosyltransferase expression levels, further reducing the definitive conclusions that can be drawn from this analysis. Despite the advancements made in studying media's

effect(s) on glycosylation, knowledge of the field is not strong enough to go beyond qualitative principles to predict precise glycosylation effects from the addition of specific media changes.

2.3 Overview of *In Silico* Glycosylation Modeling

As our paper contains the development of an *in silico* model to study glycosylation, this section is entirely dedicated to giving the reader an overview of previously developed glycosylation models (Galleguillos et al., 2017). This paper will explain the necessary assumptions, mathematical foundation, results, advantages, and disadvantages of all relevant glycosylation models. Advancement of glycosylation models can be divided into three historical time periods. The first period, from 1996-2005, represents the initiation and development of basic glycosylation models which only attempt to describe the action of relevant glycosyltransferases in the Golgi. From 2006-2014, the field undergoes a refinement and advancement of the kinetic parameters and abstraction techniques, allowing these models to incorporate cell growth and basic co-substrate formation into model abstraction. The current models (2015-2017) offer further expansion of glycosylation modeling to include the full central carbon metabolism and antibody formation in an attempt to create a dynamic glycosylation model that spans all relevant time scales.

2.3.1 First Attempts (1996-2005)

2.3.1.1 Shelikoff et al., 1996

N-glycosylation involves a sequential breaking and forming of glycolidic bonds to modify a glycan's saccharide composition through the glycosyltransferases described in the previous chapter. The consistency of the initial glycan linkage to the

protein and subsequent modifications are known as macroheterogeneity and microheterogeneity, respectively. The major factors influencing macroheterogeneity such as amino acid sequence, protein folding, and enzyme activity were identified and explored via a model built by Shelikoff et al., which is regarded as one of the first models attempting to mathematically describe the process of glycosylation.

Their model defined the system as an unfolded protein with a single glycosylation site leaving the ribosome and entering the endoplasmic reticulum (ER) at any given translocation velocity (v). They used prolactin, which only has one glycosylation site, as the model protein. Macroheterogeneity was described as the fraction of secreted proteins that are glycosylated (fractional occupancy, f). This system was modeled as a plug-flow reactor (PFR), with the residence time (τ_{gly}) represented as the length of the reactor z_{gly} over v as shown in Equation (2.1).

$$\tau_{gly} = \frac{z_{gly}}{v} \quad (2.1)$$

The PFR comprises three compartments. The first and third compartment are positions along the ER where glycosylation is not possible due to the protein being too far from an oligosaccharyltransferase, referred to as the exclusion region (z_{gly}^{excl}). However, the second compartment is free for glycosylation to take place, providing there is no competition. This compartment represents the start of the PFR and is where v and z_{gly} are initiated and set to zero. The initial model established standard kinetic rate laws for PFRs that described the glycosylation rate ($R_{Gly}(x_A)$) as a function of the fraction of aglycosylated sites (x_A) and the number of initial asparagine residues (N_0) as shown in Equation 2.2. Here, r_{char} is the characteristic reaction rate of the system in the absence of x_A

$$\frac{r_{char}}{N_0} dz = \frac{-1}{R_{gly}(x_A)} dx_A \quad (2.2)$$

From this kinetic foundation, the dimensionless Damkohler number (Da_{gly}) was established that describes the change in glycosylated asparagine site flux over the total number of asparagine sites. Additionally, they defined fractional occupancy (f) as a function of the Damkohler number as shown in Equations 2.3 and 2.4.

$$Da_{gly} = \frac{r_{char} z_{gly}}{N_0} \quad (2.3)$$

$$f = 1 - \exp(-Da_{gly}) \quad (2.4)$$

The model was used to study four unique scenarios for glycosylation initiation based on unique spatio-temporal relationships between the folding region (z_{gly}^{excl}) and the PFR segment where glycosylation can initiate. There are three kinetic states in the model. The first is when glycosylation cannot occur due to folding. The second is when no folding occurs and glycosylation can initiate uninhibited. The final state describes when folding does not completely inhibit glycosylation, and so there is a kinetic competition between the folding event and the glycosylation event. It was then used to study different permutations of these kinetic relationships along the length of the reactor, which mimics the effect of protein folding, enzyme activity, folding competition, and protein production rate on the fractional conversion of free asparagine sites. Model analysis revealed a number of trends concerning glycosylation microheterogeneity. The fractional conversion was found to be 30% higher under conditions where no folding competition existed and the folding time was minimized. The lowest fractional conversion values were reported at limiting cases of Da when folding competed with glycosylation for the entire length of the reactor.

There were a number of limitations placed on the model due to the nature of the assumptions. The ability to glycosylate a protein was modeled as a binary phenomenon, where binding is enabled if the protein is within that region but disabled

if the protein is outside that region. The more realistic scenario is that the binding kinetics vary as a function of distance from the transferase itself. Additionally, the equations only take into account a protein with only one glycosylation site. Various proteins important for monoclonal antibody manufacturing, such as EPO and immunoglobulin-G (IgG), have more than one glycosylation site which would require a modification of the kinetics in this model.

Despite these limitations, Shelikoff et al.'s model can be applied to multiple glycosylation systems due to its dimensionless nature. It was also able to thoroughly explore different kinetic assumptions to best describe glycosylation. This model is not meant to be used as a tool for glycosylation engineering, rather it is a framework designed to be refined over time to better understand and study glycosylation macroheterogeneity.

2.3.1.2 Monica et al., 1997

This paper creates a kinetic model for glycosylation microheterogeneity to describe the extent of glycan sialylation in CHO cells. As their model only attempts to describe the action of a single glycosyltransferase (sialyltransferase), their analysis is limited to the TGN, which holds the majority of intra-Golgi sialyltransferase enzymes. This paper used their kinetic model to determine the extent of sialylation in limiting cases of both sialyltransferase concentration and glycan flux into the Golgi.

Monica et al. chose to establish their system as a single *trans*-Golgi compartment, as previously stated. Based on the argument that the vesicular transport rate of proteins between Golgi compartments is orders of magnitude higher than the

residence time of a protein in the Golgi, the flux of glycans into and out of the TGN can be treated as continuous. Second, they assumed that the trans-Golgi stack could be considered a homogeneous reaction environment, as the ratio of the reaction rate including diffusivity to the reaction rate excluding diffusivity ranged from 0.93 to 0.98 over a 50-fold range of protein concentrations. Any inhibition or competition by the product glycans were negated. Lastly, all glycans entering the system were assumed to be compatible with sialyltransferase (i.e. fully galactosylated branches), and only one sialylation event was allowed per glycan.

Michaelis-Menton kinetics were used to describe the enzyme-substrate binding event that occurs when sialyltransferase modifies an eligible glycan. The results describe the rate of change of the glycan substrate and product ($d[S_1]/dt$, $d[P_1]/dt$) as a function of initial substrate and enzyme concentrations ($[S_0]$, $[E_T]$), kinetic constants for binding and dissociation (K_m , k_{cat}), and mass flowrate of the feed and exit streams (D_0 , D_1) shown in Equations 2.5 and 2.6. This equation was rearranged to express the extent of reaction ($x = [P_1]/[S_0]$) as a function of the ratio of reaction rate to transport rate ($\beta = k_{cat}[E_T]/K_mD_0$) and the ratio of substrate concentration to the kinetic constant for sialyltransferase ($\chi = [S_1]/K_m$) (Equation 2.7).

$$\frac{d[S_1]}{dt} = D_0[S_0] - D_1[S_1] - \frac{k_{cat}[E_T][S_1]}{K_m + [S_1]} \quad (2.5)$$

$$\frac{d[P_1]}{dt} = \frac{k_{cat}[E_T][S_1]}{(K_m + [S_1]) - D_1[P_1]} \quad (2.6)$$

$$x = \frac{\beta}{1 + \beta + \chi} \quad (2.7)$$

The model was used to plot the extent of reaction (x) as a function of β for varying values of χ . For the case of enzyme saturation (i.e. $\beta \gg 1$), the extent of reaction is independent of initial substrate concentration, which is included in χ . For

cases where $\beta \ll 1$, the extent of reaction is dictated entirely by the starting substrate concentration. For $\beta \gg 1$, it was found that the extent of reaction ranged from 0.1 to 1 depending on the value of χ . However, for $\beta \ll 1$, the extent of reaction only ranged from 0 to 0.1 for the same tested values of χ . This suggests that the controllability of sialylation is less affected by protein production rate when the concentration of sialyltransferase is high.

The model provides insight into factors affecting sialylation heterogeneity; however, there are key shortcomings in the analysis. The first limitation is that the model is only applicable to, at most, two competing enzymes. Second, it cannot distinguish between multiple branches on a single glycan that can become sialylated. Lastly, it does not include the glycosyltransferase's dependency on sugar co-substrates. The paper argues that the concentration of sialyltransferase co-substrate, cytidine monophosphate N-acetylneuraminic acid (CMP-NeuAc), is far more abundant than sialyltransferase. However, this assumption may not hold true for different cell lines, nor would it hold true for all glycosyltransferases should the model be expanded. Overall, Monica et al. constructed a useful model to gain qualitative insight into which conditions are optimal for sialylation and provided a flexible kinetic framework with which to apply to the rest of the glycosyltransferases.

2.3.1.3 Umaña et al., 1997

The first kinetic model that described the simultaneous action of multiple glycosyltransferases was created by Umaña et al. The model comprised 33 reactions catalyzed by eight different glycosyltransferases (ManI, ManII, GnTI-V, and GalT).

The paper also used a comprehensive literature analysis to approximate the kinetic parameters for all glycosyltransferases. This model laid the groundwork for future glycosylation modeling, and was the first to attempt to quantitatively predict glycosylation pattern trends through kinetic modeling.

The system developed for this model was unlike any glycosylation models previously, as they modeled the flux of glycoproteins through multiple Golgi compartments. This was accomplished by approximating the Golgi as a series of continuous-stirred tank reactors (CSTRs), with M9 glycans entering the system in the first CSTR (representing the *cis*-Golgi) and traveling through the *medial*, *trans*-Golgi, and trans-Golgi network (each represented by their own CSTR) to leave the system. The residence time for each CSTR was calculated by dividing the volume for each Golgi cisternae by the average protein transport rate through the Golgi. Protein transport through the Golgi was approximated as bulk flow through vesicular transport, similar to the rationale adopted by Monica et al.'s model. Each compartment of the CSTR was populated with glycosyltransferases, which dictate what reactions can take place. These glycosyltransferases were chosen because they represent major processing changes undergone by a glycan in the Golgi for N-glycosylation. The spatial distribution of the glycosyltransferases were approximated via immunoelectron microscopy studies in the Golgi. Lastly, the necessity of co-substrates was neglected in this model.

Michaelis-Menton kinetic equations were used to model all reactions. Each kinetic equation (2.8) describes glycan i in compartment j ($x_{i,j}$) for reaction m . The parameters in each equation consist of the fraction of glycan i in the previous compartment ($x_{i,j-1}$), the maximal reaction velocity for enzyme k and equilibrium

constant ($v_{m, i}$, $K_{m, i}$), and the fraction of enzyme k in the current Golgi compartment ($e_{k, j}$). This series of equations was solved using the pseudo steady-state approximation to convert all ordinary differential equations into nonlinear algebraic equations. The algebraic equations were solved iteratively.

$$x_{i,j-1} = x_{i,j} + \frac{v_{m,k} e_{k,j} x_{i,j}}{K_{m,i} + x_{i,j}} \quad (2.8)$$

The kinetic parameters estimated from literature were representative of free glycan substrates *in vitro*. Consequently, they are unlikely to match the glycoform produced by CHO cells *in vivo*. However, the paper compared the glycoform produced by the model using the basal parameter set with a glycoform found in typical CHO cell glycoproteins to qualitatively evaluate model accuracy. The distribution created by the model was qualitatively similar to wild-type CHO glycoforms, with the majority of glycans secreted as biantennary and triantennary complex glycans. They then altered the protein production rate to understand how the glycoform responds to an increased flux of glycans. They found that as the flux of glycoprotein increases, glycans gradually become less processed, and higher fractions of high mannose forms are observed. An increase of protein productivity from 500 to 2000 pmol/10⁶ cells/day reduced the percentage of biantennary glycans by 25%. Lastly, the activity of GnTIII was increased in the model to simulate the effects of GnTIII overexpression. This was found to reduce the fraction of bisected complex glycans by over 30% as well as increase the fraction of bisected hybrid glycans by over 50% due to substrate competition between ManII and GnTII.

Some key shortcomings to the model are the low number of glycosyltransferases included, the lack of co-substrate requirements in the kinetic equations, and the numerous assumptions made during parameter estimation. Though

the paper asserts that their parameters were within a ‘reasonable range’ for model performance, it cannot quantitatively match the many unique experimental glycoforms found in literature. However, the model was able to capture qualitative trends in glycosylation and highlight competitive behavior between glycosyltransferases.

2.3.1.4 Kawano et al., 2005

The model from Kawano et al. is not a kinetic model, rather it uses statistical methods to evaluate the likelihood that a particular glycoform can be achieved given a set of glycosyltransferase microarray expression data. It is the first model that attempts to link the glycosyltransferase transcriptome with mathematical machinery that predicts human-cell glycosylation patterns. The model was trained on data from the KEGG glycan database and tested on expression profiles from human cells. The accuracy of the model surpassed 81% on the training data, and the model was able to correctly predict a glycan distribution representative of carcinoma cells based on the cells’ transcriptome.

Kawano et al.’s model does not mimic any physical biological entity, such as the Golgi. Instead, the model consists simply of a reaction library for all reactions catalyzed by 98 glycosyltransferases from the human genome. Rather than reactions consisting of glycan substrate and glycan product, each reaction consists only of the (terminal) acceptor saccharide residue, the donor saccharide residue, and the type of linkage between them. In this way, an otherwise enormous reaction network is distilled into only 42 reaction patterns. 4107 glycans were included in the model from the KEGG database. A co-occurrence score between all reaction patterns was

calculated that gave a measure of the likelihood that two reactions would occur consecutively. Reactions closely related in terms of substrate type or location exhibited a high score, while distant reactions exhibited a low score.

To predict a certain glycan, the glycan was converted into a reaction pattern sequence that allowed any glycan to be compared with the reaction library. The co-occurrence scores for both the query glycan and the reaction pattern library calculated from expression data were compared to determine how likely the query glycan would be observed in experiment. Using this system, the model was able to achieve an accuracy (percentage of glycans correctly predicted in the model's top 10) of 81%, which is reasonable given that this model contains no kinetic data.

The model does suffer from a number of drawbacks which keep it from providing industry and academia with an accurate, useful glycosylation model. The primary weakness is that the top 10 glycans most likely to appear as predicted by the model given a set of glycosyltransferase expression data are binary. This means that there is no quantitative relationship between the glycans present in the top 10 ranking. This can pose problems when evaluating if a particular glycan will constitute a major or minor part of the glycoform. Secondly, the predictive accuracy of the model is reliant on microarray expression data of all glycosyltransferases, which is not always available. Lastly, the model cannot take into account the presence of all sugar co-substrates required for the successful action of glycosyltransferases.

2.3.1.5 Krambeck & Betenbaugh, 2005

The model created by Krambeck & Betenbaugh was the first model to completely describe all major CHO cell N-glycosylation pathways. This model is an extension of Umaña et al.'s previous work on glycosylation modeling, and includes all 11 major N-glycosyltransferases. Additionally, this model is the first to include the requirement for sugar co-substrates in all glycosylation reactions. Over 7,000 glycans and 22,000 reactions were included in the model. Additionally, this work included an algorithm to fit their model to any experimental glycoform by adjusting the concentration of all enzymes to minimize error between the model and experimental glycoforms. Literature was used to find or estimate the values for all kinetic parameters. The model was applied to determine quantitative effects of increased glycoprotein production on the glycoform. Lastly, Krambeck and Betenbaugh used their model to identify optimal cellular engineering strategies that could maintain a consistent glycoform after increased protein production.

The Krambeck model is kinetic, and defines the Golgi as a series of CSTRs in a similar way to the Umaña model. The parameters for transport speed, Golgi volume, and compartment volume were calculated from literature. A key improvement over the Umaña model is the expansion of the reaction network to include reactions catalyzed by all major N-glycosylation reactions (ManI, ManII, GnTI-V, GnTE, FucT, GalT, and SialT) which results in a network orders of magnitude larger than that created by the Umaña model. The inclusion of more glycosyltransferases meant more kinetic parameters were needed to run the model. However, multiple kinetic parameters were unable to be found, and so they established a series of assumptions for parameter estimation. For example, the enzyme concentrations for FucT, GnTE, and SialT were

set to an arbitrary level within the range of other known glycosyltransferase concentrations (such as GnTIII and GnTIV) due to lack of data. Additionally, because the equilibrium constants for sugar co-substrate-binding to galactosyltransferase was unknown, the enzyme was assumed to be saturated with co-substrate. The model takes the basic M9 glycan in the *cis*-Golgi and uses the established kinetic reaction network to solve for the final distribution of glycans secreted from the TGN. The reactions were described with Michaelis-Menton kinetics. One key addition made in the Krambeck model is the inclusion of sugar co-substrates in the kinetic equations (Equation 2.9). The final rate equation includes variables for the forward and reverse rate coefficients (k_f and k_r), the dissociation constants for both substrates (K_{mi} and K_{md}), the apparent overall equilibrium constant (K'_{eq}), the concentrations of substrate and product ($[P_i]$ and $[P_{i+1}]$), concentration of enzyme in compartment f ($[E_f]$), and the concentrations of the co-substrate and residual UDP sugar ($[UDP_S]$ and $[UDP]$). This is a necessary first step towards a truly complete model for glycosylation. However, the increased complexity of this model requires more assumptions for operation. Concentrations of all sugar co-substrates (UDP-GlcNAc, UDP-Gal, CMP-NeuAc, and GDP-Fuc) would need to be estimated within each compartment of the Golgi. The model used data for sugar co-substrate abundance in the cytoplasm of baby-hamster kidney (BHK) cells as a surrogate for CHO cell co-substrate abundance.

$$r = \frac{k_f[E_f]([UDP_S][P_i] - \frac{1}{K'_{eq}[P_{i+1}][UDP]})}{K_{mn}(K_{md} + [UDP_S])(1 + \sum_j \frac{[P_j]}{K_{mj}}} \quad (2.9)$$

Krambeck et al. compared their model's performance with the Umaña version to identify strengths and weaknesses of the added functionality. They found that the fraction of biantennary glycans increased by 5% when removing Umaña's neglect

of co-substrates, indicating that sugar co-substrate availability is more important for glycosylation microheterogeneity than previously assumed. Next, they compared the Krambeck model with experimental data to see if it could accurately reproduce a real glycoform. The basal parameter set was not able to accurately recreate the chosen glycoform, with a variability in the biantennary sialylated glycan of 50%. Krambeck et al. postulated that the many assumptions used in estimating glycosylation parameters along with the extent of variability in glycosylation over a cell culture's lifetime warranted the creation of an algorithm to fit their model to experimental data. This was accomplished by changing the glycosyltransferase concentrations in each Golgi compartment to achieve the best fit between model and experiment. After fitting, all predicted glycans were within 2% of the levels measured in the corresponding experiment.

Krambeck et al. lastly used their model to aid in glycoform optimization for CHO cells. Industry is constantly seeking to increase protein production, as more protein results in a greater number of sales. However, Umaña's model projected that a large enough increase in protein production would alter the glycoform, potentially compromising product quality. Krambeck et al. addressed this concern by using their model to identify the best way to maintain glycosylation consistency with increased protein production. First, the total glycan concentration was increased four-fold (500-2000 μ M), simulating a dramatic shift in protein production rate. The model projected that increasing the glycan concentration decreased the average antennarity of all glycans from approximately 3.05 to 2.78 antennae/glycan. The percentage of sialylated glycans also decreased from approximately 60% to 31%. Next, the high-productivity model was fit to the original glycoform to determine which

glycosyltransferase concentrations needed to be modified to restore the glycoform from the low protein production case. They found that increasing the concentration of GnTV and SialT by factors of 2.7 and 1.8 respectively was required to recover the original glycoform.

The Krambeck model was a landmark achievement, giving us a tool to study the interaction of most major components of glycosylation for the first time. However, the assumptions put in place for various kinetic parameters made a fitting algorithm necessary for useful model performance. Additionally, the model cannot bridge key parameters in the culturing environment that affect glycosylation such as ammonium concentration, glucose levels, and sugar co-substrate feeding. This makes the model unwieldy in an industrial context, where the optimum culturing conditions need to be established for consistent and satisfactory product quality.

2.3.2 Model Advancements (2006-2014)

2.3.2.1 Hossler et al., 2007

The model developed by Hossler et al. is a kinetic model that aims to understand more about the mechanism of protein transport through the Golgi and its effect on glycosylation. There are two proposed mechanisms that explain intra-Golgi protein transport: vesicular transport and Golgi maturation. Vesicular transport involves the transfer of proteins through each Golgi stack through vesicles, while

cisternal maturation proposes that the protein remains in the same Golgi stack until secretion, and the enzymatic composition of the cisternae is changing (Chapter 2.2.2). Mathematical abstraction is unique for each case. Vesicular transport is modeled as a series of CSTRs, consistent with previously developed glycosylation models. Protein transport through cisternal maturation, on the other hand, is best represented as substrate traveling through a PFR (Equation 2.10). In this equation, for n compartments, C^n represents the concentration vector for all N-glycans, R is the reaction rate vector, v is the linear velocity of flow through the PFR, D is the axial diffusivity across the PFR, and S is the stoichiometric matrix for all reactions. This model compares glycan distribution for each case. These systems were extensively studied under different glycosyltransferase distributions to better understand variations in glycosylation microheterogeneity. This model is not meant to be used for glycosylation prediction, rather it is a tool for learning more about how different methods of abstraction in protein transport affect glycosylation.

$$\frac{dC^n}{dt} = -v \frac{dC^n}{dz} + D \frac{d^2 C^n}{dz^2} + SR^n \quad (2.10)$$

Similar to the models developed by Umaña and Krambeck, this model is a kinetic representation of glycosylation. An N-glycosylation reaction network is generated describing the path of the basic M9 glycan through all four compartments of the Golgi (*cis*, *medial*, *trans*, and *TGN*). All compartments were assumed to have equal volume, in line with the assumptions made in previous glycosylation models. The reactions included in the network were those catalyzed by 10 major glycosyltransferases (ManI, ManII, GnTI-V, FucT, GalT, and SialT). To simplify the problem, no retrograde flow of glycoproteins was considered and the model was established as describing purely anterograde transport. Literature was used to establish

the distribution of all glycosyltransferases in each Golgi compartment. As previously mentioned, the method of transport through the Golgi was set to either a series of CSTRs (vesicular transport theory) or a PFR (cisternal maturation theory). The mathematical difference between these two abstractions is due to the definition of residence time in each Golgi compartment. For the CSTR model, there is a distribution of residence times in each compartment, and two glycans within the same compartment can theoretically move on to the next compartment at different times. For the PFR model, glycans are only allowed to move onto the next compartment after reaching the distal end of their current compartment. This implies a single residence time value for all glycans at a certain time point. Just as glycans would be confined to a single Golgi compartment in cisternal maturation theory, the PFR model confines all glycans that entered the reactor together to the same residence time. All kinetic parameters for the enzymes were taken from literature, and co-substrate competition was included in the model. As this model is meant to study specific attributes of glycosylation, the concentrations of both enzymes and sugar co-substrates were established as to create an even distribution of all terminal glycosyltransferases, so that the effect(s) of system disturbances can be easily observed.

The initial PFR model was first used to identify the residence time required to completely process all glycans. It is important to note that for these tests, only a single Golgi compartment was included in the PFR model, and all glycosyltransferases were populated in the same compartment. As expected, they found that the complexity of all glycans gradually increased as the time spent in the reactor increased. It took approximately 40 min in the PFR to completely process all glycans to a terminal (fully sialylated) state. A sensitivity analysis was then performed to identify which

glycosyltransferases had the most severe effect on the glycoform. The sensitivity coefficient for glycan i (δ_i) is described by the flux of that N-glycan exiting the Golgi (J_i), and the concentration of glycan i in the Golgi (c_i). They found glycosyltransferases that regulated specific branching points in the glycosylation pathway, such as GnTII, GnTIII, and ManII, had the highest sensitivity coefficients (Equation 2.11) across a wide range of glycans.

$$\delta_i = \frac{c_i}{J_i} \frac{dJ_i}{dc_i} \quad (2.11)$$

Next, they attempted to modify all glycosyltransferase concentrations to achieve a single glycan as the output. They attempted this for each terminal (fully sialylated) glycan in their model and found that terminal glycans previously acted on by GnTIII, GnTIV, and GnTV were impossible to isolate. The reason for this is due to the enzymatic competition and substrate specificity between these three glycosyltransferases.

Upon examining the effects of splitting the PFR into four compartments with an equal residence time (10 min. each), they found a decrease in terminally-processed glycans and a concurrent increase in immature glycans. The reason for this was due to inefficient holding times in the initial compartments. If a glycan did not reach the adequate stage of processing by the time it was removed from the first compartment, substrate specificity prohibits that glycan from further processing. The final result obtained in this study was a comparison of the glycoforms produced by a PFR reactor scheme and a CSTR reactor scheme, both having the same total residence time. This test was run for residence times of 40 min and 20 min. They found that for 40 minutes, the heterogeneity expressed in the CSTR system is higher than for the PFR system. This effect was magnified when the residence time for both systems was reduced to 20

minutes. The CSTR system exhibited a large variation in the processing level of glycans in the glycoform, while the PFR system had a majority of high mannose glycans.

This model explored the effect(s) of residence time and Golgi abstraction on glycosylation to determine which settings were the most realistic to use when predicting real glycoforms. The residence time in the Golgi was found to be an important variable, heavily dictating the heterogeneity observed in glycoforms for both the PFR and CSTR models. PFR compartmentalization was also found to have a dramatic effect, as glycosyltransferase localization prevented a portion of the glycoform from being terminally processed. Lastly, when comparing the PFR and CSTR models, the PFR model tends to produce immature glycans in higher abundance than the CSTR model. This is somewhat reflective of experimentally-derived glycoforms, but the evidence is not strong enough to definitively claim one method is superior over the other. Despite this, the paper offers a way to study and compare protein transport and glycosylation in the Golgi of mammalian cells.

2.3.2.2 Kontoravdi et al., 2006

The model created by Kontoravdi et al. was the first to link cellular growth dynamics, mAb production, and glucose metabolism to glycosylation. The model was validated by successfully replicating cell growth of hybridoma cells. An optimization analysis was performed to explore the effect of feeding schedule changes on mAb production and quality. This analysis was used to guide cell engineering towards more

efficient mAb production and represents a first step towards a model that can unify culturing conditions and glycosylation.

This model links a series of separate models together to map a culture of cells through the growth phase, calculate the resulting mAb production and secretion rates, and determine the glycosylation patterns from sugar uptake rates and glycosyltransferase activity. Cell growth was represented with Monod kinetics using glucose/glutamine uptake and ammonia concentration to predict growth and death, respectively. The cellular metabolism equations were represented as simple mass balances for glucose and glutamine. To model mAb production, Kontoravdi mapped the formation of a single mAb from two heavy chains (H_2) and two light chains (L_2). Specific kinetic constants (K_{ER} and K_G) describe mAb transport into the ER and Golgi respectively (Equation 2.12). An efficiency factor (ε_I) is also established to control how many mAbs become glycosylated.

$$\frac{d[H_2L_2]_G}{dt} = \varepsilon_I K_{ER} [H_2L_2]_{ER} - K_G [H_2L_2]_G \quad (2.12)$$

In this model, the reactions included were identical to those from the Umaña model. All constants corresponding to the kinetic equations in the reactions were taken from Umaña's work. However, they modified each equation to include competitive inhibition from product mAbs, which was neglected in previous models (Equation 2.13). Here, for i glycans, k reactions, l reactions catalyzed by a single enzyme, and j compartments, the model's rate equations contain information for the maximum reaction velocity ($v_{k,j}$), the volume of culture (V), volume of compartment j (V_j), concentration of viable cells (X_V), fraction of glycan i in compartment j ($x_{i,j}$), dissociation constants for reaction k ($K_{l,k,j}$), amount of glycoprotein in compartment j (p_j), and the volume of Golgi compartment j (V_j). It is important to note that as this

model is built off of Umaña's kinetic interpretation of glycosylation, it does not require the presence of sugar co-substrates.

$$r_{k,j} = \frac{v_{k,j} X_V p_j x_{i,j}}{p_j x_{i,j} + K_{l,k,j} V_{G,j} [1 + \frac{p_j}{V_{G,j}} \sum_l (\frac{x_{l,j}}{K_{l,l,j}})]} \quad (2.13)$$

The parameters concerning cell growth and mAb production kinetics were determined by calibrating their model to literature. Once calibrated, the model correctly predicted the total mAb concentration over the course of a 200 hr. run, and was able to correctly achieve the 26 mg/L titer that was observed at the end of the experiment. However, the model underpredicted the glucose concentration of the culture from hours 20-60 and overpredicted the final concentration by approximately 2 mM at 120 hrs. Glycosylation patterns were qualitatively analyzed and appeared to be within range of typical hybridoma glycoforms, as the glycoform consisted of high mannose, biantennary, and bisected glycans. The model was applied to examine optimal feeding schedules for fed-batch production of mAbs, taking both production rate and product quality into consideration. The feeding schedules were varied by modifying the number of feeding intervals in a 250 hr. period (analyzing 1, 3, 4, 5, and 10 intervals within this period). Inlet flowrate, glucose levels, and glutamine levels were optimized for each feeding schedule to obtain the greatest mAb production. This was used to identify a feeding interval of 0.1 hr. as the optimum schedule for high mAb production. By tracking the projected glycoform for each schedule, glycans with bisecting GlcNAc residues changed by approximately 5% between the 1 and 10-interval cases. This analysis was used to demonstrate the advantages of applying *in silico* modeling to cell culture optimization.

Though the model can accurately predict mAb production rate and glycosylation simultaneously, it still does not include the variance introduced by sugar

co-substrates essential for glycosylation. It also does not include a significant portion of N-glycans created by the enzymes SialT and FucT, limiting its predictive strength. Lastly, as the quality of glycoform data in literature was poor, the glycosylation portion of their model could not be quantitatively validated. However, as techniques for glycosylation monitoring are improved, the model can act as a suitable foundation for a comprehensive model that includes both cell growth, mAb production, and glycosylation.

2.3.2.3 Senger & Karim, 2008

Senger & Karim's model used machine learning and artificial neural networks to predict the primary glycan type (high mannose or complex) as a function of glycoprotein secondary structure characteristics surrounding the glycosylation site. The model was trained with experimental data to predict the most probable protein secondary structure from the amino acid sequence. After training the model, they found that glycoprotein secondary structure was a strong predictor of glycan type, and the model was used to successfully study the effects of amino acid sequence of glycosylation.

When gathering training data for the model, 158 protein sequences along with the most abundant glycan type (high mannose vs complex) were used as training metrics. The neural networks were used to connect the protein's amino acid to a secondary structure, and this information was used to predict the glycan type of the reference data set. Cross-validation was performed with the data, which involves taking 10 random data sets from the reference set to use in a neural-network training

algorithm. The selected 10 data sets are used as the testing set, and the remaining 148 data sets are used for neural-network training. This procedure is repeated for a different selection of 10 data sets many times until the optimum neural network has been achieved. The secondary structure of amino acids surrounding the glycosylation site was classified as helix, extended, or unordered. The effectiveness of prediction for the testing sets was dictated by the model's mean-squared error (MSE). As the prediction results are only binary (it only predicts the glycan type of the majority, not the percentage of the majority in the glycoform), the MSE was calculated from the percentage of correct predictions out of the testing set.

First, the model was tested under multiple amino acid combinations surrounding the glycosylation site to determine the correlation between amino acid sequence, solute accessibility, and glycan type. The amino acid search was limited to 10 residues on either side of the glycosylation site. No clear trends could be found, and they hypothesized that the glycosylation type is likely due to the combined effects of secondary structure and solute accessibility. After this analysis, training of the neural network began using the cross-validation procedure described previously. They found that prediction results improved as more surrounding amino acids were considered. However, no global optimum was reached, and more amino acids would need to be searched in order to confirm the optimum number of surrounding amino acids to study. The optimum MSE achieved by the model training was 79%.

Lastly, the model was tested on a series of tissue-type plasminogen activator (tPA) mutant clones. Each mutant had a specific amino acid alteration in the sequence surrounding their glycosylation site N117. Additionally, the primary sequence input data was removed from the model, and only the secondary-structure data was used as

the model's predictor. Six out of eight mutations were correctly predicted, and the incorrect predictions had confidence scores below 65% and could be easily identified. This model demonstrates the importance of amino acid structure surrounding the glycosylation site on glycosylation. However, as the model does not predict all glycans in the glycoform, its utility from a glycosylation-engineering perspective is limited.

2.3.2.4 Krambeck et al., 2009

Quantifying a glycoform experimentally is a tedious process, and involves the accurate interpretation of high-quality MALDI mass spectrometry (MS) data of the glycoform. However, as many glycans occupy the same position on MS output, accurate interpretation of MS data is difficult. This paper expanded the original Krambeck glycosylation model to produce a synthetic mass spectrum associated with the glycoform produced from their model. This mass spectra can then be compared with the mass spectra from the experiment, and the enzyme concentrations are adjusted to match the synthetic and real spectra with high accuracy. In this way, the model can recreate the mass spectra from the bottom-up, and the complex kinetic interplay of all glycosyltransferases can be captured in the MS analysis. The model was validated through analysis of monocytic leukemia cells to determine the differences in glycosylation patterns developed in the disease state.

The base structure of the model was identical to the Krambeck model developed in 2005. As a result, the kinetic and transport parameters, mathematical abstraction of protein transport through the Golgi, and solving algorithms were all

taken directly from previous work. However, Krambeck et al. made dramatic alterations to the glycan and glycosyltransferase database to allow for an expanded network of potential glycan structures. In the 2009 Krambeck model, stereoisomers (i.e. glycans with the same chemical bond structure but different bond types) were considered indistinguishable. This assumption was lifted in the current model, which results in an additional set of glycosyltransferases that can be included in the model. For example, there are two types of sialic acid linkages that can occur for N-glycans in CHO cells (an $\alpha(2, 3)$ or an $\alpha(1, 6)$ linkage). These linkages are both catalyzed by different sub-groups of sialyltransferases, which this model now recognizes as distinct. Additionally, a new algorithm for glycan annotation was developed to make glycan generation and modification easier and more comprehensive. The algorithm was capped at structures with a molecular mass of 4000 g/mol, as that represents the upper range of molecular weights capable of being measured by MS. Lastly, an algorithm was developed that replicates a hypothetical mass spectrum based on the glycoform output from the model. In this way, rather than calibrating the glycoform to experimental data, they calibrate the model to MS data. Care was taken to process the experimental MS data correctly for proper comparison to the *in silico* MS data.

The model was tested with a diseased line of leukemia cells for validation. The calibration to available MS data resulted in strong agreement between the model and experiment. Though there was no reported error metric used to quantify model accuracy, the visual representation showed a strong match between the *in silico* and experimental spectra, with the highest peak inaccuracies on the order of <1%. There is a small amount of deviation for peaks of low intensity (on the order of 1%). However these do not represent a large percentage of the glycoform. The glycoform was

analyzed and it was found that leukemia cells exhibited increased concentration of the GnTIII enzyme by 3.5-fold over their healthy counterparts. The model presented here can replicate any mass spectrum of glycosylation and connect the output to the glycosyltransferase expression levels required to produce the measured glycoform. However, a number of corrections needed to be made to the fitting algorithm to fit the model accurately, limiting its ability to predict glycosylation patterns without an experimental mass spectrum.

2.3.2.5 Jimenez et al., 2011

Then model created by Jimenez et al. was the first complete kinetic model to describe glycan transport through the Golgi using the cisternal maturation theory, discussed previously in Hossler et al. (2007). While Hossler et al. created a glycosylation model with only a fraction of the required glycosyltransferases as a means to study the effect of cisternal maturation on glycosylation, this work includes all main glycosyltransferases and sugar co-substrates as well as optimization algorithms to predict experimental glycosylation patterns. The kinetic parameters for all glycosyltransferases and co-substrates as well as residence times for each Golgi compartment was estimated according to literature. The model was used to replicate the glycoform of IgG-producing CHO cells as well as correctly predict the effects of a *FUT8* knockout on glycosylation.

A number of distinctions separate this model from its predecessors. As previously explained, the Golgi is abstracted as a PFR with changing enzyme concentrations along its length rather than a series of CSTRs. This modifies the

residence time distribution of glycans moving through the reactor, decreasing glycosyltransferase heterogeneity for short residence times compared to the CSTR interpretation. The concentration of glycosyltransferases along the length of the PFR are defined by three-parameter normal distribution functions. The equations used to describe the kinetics were more sophisticated than basic Michaelis-Menton relationships. Each reaction was modified based on the appropriate product competition and equilibrium relationships, such as sequential-order bi-bi kinetics (Equation 2.14) and random-order bi-bi kinetics (Equation 2.15). These equations describe the relationship between the concentration of substrate ($[OS]$), sugar precursor ($[PC]$), and enzyme ($[E]$) for glycan i , product $i+1$, enzyme j , precursor k , and competing glycans z .

$$r_j = \frac{k_{f,j}[E_j][PC_k][OS_i]}{K_{d,i}K_{d,k}\left(1 + \frac{[PC_k]}{K_{d,k}} + \frac{[PC_k][OS_i]}{K_{d,k}K_{d,i}} + \frac{[PC_k]}{K_{d,k}} \sum_{z=1}^{\infty} \left(\frac{[OS_z]}{K_{d,z}} \right) + \frac{[B_k][OS_{i+1}]}{K_{d,Bk}K_{d,i+1}} + \frac{[B_k]}{K_{d,i+1}} \right)} \quad (2.14)$$

$$r_j = \frac{k_{f,j}[E_j][PC_k][OS_i]}{K_{d,i}K_{d,k}\left(1 + \frac{[PC_k]}{K_{d,k}} + \frac{[OS_i]}{K_{d,i}} + \sum_{z=1}^{\infty} \left(\frac{[OS_z]}{K_{d,z}} \right) + \frac{[PC_k][OS_i]}{K_{d,k}K_{d,i}} + \frac{[PC_k]}{K_{d,k}} \sum_{z=1}^{\infty} \left(\frac{[OS_z]}{K_{d,z}} \right) + \frac{[B_k][OS_{i+1}]}{K_{d,Bk}K_{d,i+1}} + \frac{[B_k]}{K_{d,i+1}} + \frac{[OS_{i+1}]}{K_{d,i+1}} \right)} \quad (2.15)$$

The majority of kinetic parameters were estimated from literature. However, a parameter estimation algorithm was used as a means to establish a basal parameter set for all unknown values. The same number of glycosyltransferases were modeled in this work as those in Krambeck and Betenbaugh's model (2005). The fractional distribution of enzymes was taken from literature and converted into a normal distribution for each glycosyltransferase. The absolute values of all glycosyltransferase concentrations were calculated such that all input glycans were terminally processed with 50% sialylation. Lastly, kinetic values for co-substrate transport into the Golgi were estimated by making the assumption that the rate of accumulation for each co-substrate was equal to the rate of by-product accumulation

in the cell. This allowed them to estimate basic kinetic parameters for all relevant processes in the Golgi.

The basal parameter set was compared against a replicated version of the Krambeck 2005 and Hossler 2007 models as each one attempted to match the glycoform of a Herceptin-producing cell line. Model operation was similar to Krambeck's, where the parameters were fit to a particular glycoform. However, rather than modifying the concentrations of all glycosyltransferases, six dissociation constants for different glycosyltransferases were varied to fit experimental data. The model resulted in a higher accuracy than either of its two predecessors, confirming that the kinetic adjustments created a more realistic system. The system was then applied to predict the effects of a *FUT8* knockout from an IgG-producing CHO cell line. This first involved calibrating the model parameters to the experimental glycoform, as described previously. Then, the concentration of *FUT8*'s corresponding gene, FucT, was set to zero to mimic a complete gene knockout. Model predictions were highly accurate, exhibiting a deviation of only 4% from the experimental data.

This model has taken a step closer towards an accurate, unified glycosylation model by enhancing the kinetic equations and including co-substrate transport into the Golgi. Future steps for this model would be to link co-substrate production to the central-carbon metabolism. Despite its advantages, the model still needs to manipulate kinetic parameters to achieve optimal fits for various glycoforms.

2.3.2.6 St. Amand et al., 2014

This work took a unique approach to analyze kinetic glycosylation models with the goal of determining which aspects of glycosylation are intrinsically controllable. This strategy involves obtaining a process gain matrix of the system through a carefully designed set of simulations according to statistical design of experiments (DOE) and subsequent ANalysis Of VAriance (ANOVA) to determine the controllable output modes of the glycosylation network. These modes were related to the controllability of specific glycan classes to determine whether the production of a certain glycan(s) was intrinsically impossible given the starting state of the system and the controllability of the network.

Mathematically, a system of n outputs and m inputs is considered controllable if the derived controllability matrix has rank n . However, creating the controllability matrix properly for glycosylation is too difficult due to the enormous scale of the reaction network. Instead, St. Amand et al. aimed to create a process gain matrix which will serve as a map between any input vector and the appropriate output vector (Equation 2.16). This matrix (generated for i outputs and j inputs) is estimated by a statistical DOE that will relate a series of carefully selected input states to the resulting output state from the glycosylation model. The process gain matrix is then transformed back into a controllability matrix to assess controllable output modes. In this case, the input vector consists of concentration values for all 15 N-glycosylation enzymes in the Golgi. Note that this procedure is not part of the glycosylation model, rather it is a mathematical analysis performed on a pre-existing glycosylation model to assess network controllability at the micro-scale.

$$\Delta y_i = \sum_{j=1}^m (\beta_{ij} \Delta u_j) \quad (2.16)$$

The mathematical model analyzed with this technique is identical to the model developed by Krambeck & Betenbaugh in 2005. It consists of 15 glycosyltransferases and over 7,000 possible glycoforms. This model approximates the Golgi as a series of CSTRs and populates each CSTR with the distribution of glycosyltransferases estimated to reside in the corresponding Golgi compartment. All simulations used this model in the established DOE and ANOVA analysis. The DOE encompassed a resolution-IV fractional factorial design, which separates enzyme main effects from two-factor interaction effects. Two concentration values for each glycosyltransferase were used as the 'low' and 'high' factors for each range, and three ranges in total were analyzed (spanning 0.2-9.05 μM).

Analysis of the process gain matrix for the three ranges showed that various output modes were controllable in ranges 1 and 2, but no modes were controllable in range 3. This implies that the concentration levels used in range three either span too small of a range or are in a suboptimal input space to result in any significant controllability on the system. To identify which specific output modes are controllable, the singular value decomposition of each process gain matrix was calculated. For range 2, ManI, GalT, and SialT were found to have the largest number (8, 8, and 7 respectively) of highly controllable output modes (defined as an absolute value of the process gain value > 1). The next step involves relating this information back to specific glycan classes to determine which glycoforms can be created given a basic IgG starting glycoform. Each output mode was related back to the specific glycans that compose it. The galactosylated and sialylated glycans were shown to be the most controllable, which is consistent with the previous analysis that isolated GalT and SialT as possessing the most controllable output modes.

The last result presented by this paper involved assessing the controllability of achieving specific glycoforms from any arbitrary starting glycoform. A number of simple glycoforms were defined which were composed of a biantennary IgG glycoform with varying GlcNAc, galactose, and sialic acid residues. A distribution of process gains were calculated for each glycosyltransferase required to achieve that glycoform, assuming all glycans start as the basic M9 form. From the process gains, over half of the glycans were not controllable in the operating ranges tested, indicating that changing the glycoform from and to any desired state through modifying glycosyltransferase concentrations alone is intrinsically difficult. However, this difficulty involved controllability of highly-processed glycans, such as completely galactosylated and sialylated glycans. Less-processed glycans such as high mannose, and biantennary glycans with one or no galactose residues were significantly more controllable than their completely processed counterparts.

As this model is a direct copy of Krambeck & Betenbaugh's work, it suffers from the same weaknesses, including a number of arbitrarily defined kinetic parameters and an inability to relate controllability to cell culturing conditions. Work would need to be done to extend and validate Krambeck's kinetic model before expansion on the network controllability. It is likely that hurdles to process controllability exist between cell culturing conditions attainable by industry and the concentration of glycosyltransferases and/or co-substrates, decreasing the magnitude of many process gains calculated in this paper.

2.3.2.7 Jedrzejewski et al., 2014

This paper builds off of the work done by Jiminez et al. to create a model that links cellular culturing conditions, co-substrate production, and glycosyltransferase action in the Golgi. This resulted in a model that can accurately match the cellular growth rate, intracellular nucleotide concentrations, and intracellular sugar co-substrate concentrations by taking only glucose/glutamine uptake as an input. This model was linked to Jiminez et al.'s model to successfully replicate the glycoform of mouse hybridoma cells. This work allows reasonable co-substrate distributions to be predicted from only one input parameter from cell culture.

There were three mathematical sections required for model operation: quantification of cell growth/death, nucleotide synthesis, and sugar co-substrate synthesis. These pieces are linked to Jiminez et al.'s glycosylation model to recreate specific N-glycosylation patterns. The cell growth/death model uses Monod kinetics to relate the net amount of glucose/glutamine coming into and out of the system to cell growth. The nucleotide synthesis model contained balances for six nucleotide molecules essential in the synthesis of co-substrates (adenosine diphosphate, adenosine monophosphate, adenosine triphosphate, cytosine triphosphate, guanine triphosphate, and uridine diphosphate). The sugar co-substrate synthesis pathways were established through a literature comparison of murine cells. This network diverts glucose away from glycolysis and creates all co-substrates required for glycosylation. The reactions were described with Michaelis-Menton kinetics, and augmented with product competition terms depending on the reaction type similar to the equations presented in Chapter 2.3.2.5.

A number of assumptions and data taken from literature were used to establish reasonable estimates for all kinetic parameters. Simple assumptions, such as that enzyme concentrations remained constant in the sugar co-substrate metabolism, were made if data was absent. Despite this, 46 parameters could not be identified through literature alone. To solve this, a parameter estimation program was employed; this program uses global sensitivity analysis to fit all missing parameters based on a set of cellular growth data.

First, the cellular growth and metabolism elements were tested to see if an accurate representation of intra-cellular metabolic concentrations could be calculated from glucose uptake alone using mouse Murine cells. The model was able to accurately predict cell density, glucose concentration, glutamine concentration, and titer within 25% of the experimental data for all starting and final values (for a 150 hr. culture). The maximum deviation for each parameter (cell density, glucose concentration, glutamine concentration, and antibody titer) was calculated to be approximately 11%, 22%, 30%, and 21% respectively. The nucleotide sugar concentrations at the end of the culture time were incorrectly predicted by a margin of >50% for CMP-NeuAc, GDP-Fuc, and UDP-GlcNAc, with UDP-Gal exhibiting agreement between model and experiment at the end of culture. During the culture time, massive deviations were observed for GDP-Fuc (>500%), and this was attributed to the low intracellular concentration of the metabolite. The qualitative behavior of the model for all co-substrates matched the experimental data with the exception of GDP-Fuc. The sugar co-substrate transport rates into the Golgi given by the model were then used in a copy of Jiminez et al.'s model to predict the resulting glycoform exhibited by the murine cells. This procedure was able to successfully replicate the

target glycoform (consisting of three glycans) using the co-substrate concentrations provided by the model, with a maximum glycan deviation of 9%.

Despite its accurate performance, there are a number of weaknesses that prevent it from becoming a truly predictive model. The model by Jiminez et al. that was used to predict glycosylation profiles needs to calibrate its kinetic parameters to fit the target glycoform, which means it cannot predict all glycoforms *a priori*. Second, the parameters used in the model were calibrated to Murine cells, not CHO cells. As CHO cells exhibit different cellular behavior, researchers will need to modify the model to be relevant in an industrial context.

2.3.3 Current State (2015-Present)

2.3.3.1 Spahn et al., 2015

The model developed by Spahn et al. uses a constraint-based framework to study glycosylation not seen in previous models. Rather than developing or expanding on kinetic models, they use Monte Carlo sampling and Markov chain theory supplemented with flux-balance analysis to statistically predict the effect(s) of gene knockouts on CHO cell glycosylation. Due to the statistical and probabilistic nature of these techniques, the model does not require any kinetic or culturing parameters for successful operation. This means that, by using a wild-type glycoform as the only input, the model can predict quantitative effect(s) on glycosylation due to gene knockouts. This offers a number of advantages over kinetic models, including fewer

measurements, a lower number of assumptions, and a lack of reliance on kinetic parameters, many of which are approximated in current kinetic models.

Markov chain theory is a probabilistic description of a system in which a stochastic network of state transitions is established for a series of elements. These elements are connected to one another through transition probabilities, and once an element has transitioned, there is no memory of its original state. Glycosylation can be abstracted as a Markov chain, as glycans transition amongst each other with specific probabilities. These transition probabilities are related to kinetic and biological properties, such as the concentration/activity of the governing glycosyltransferases, the availability of co-substrates, and substrate competition.

To determine the transition probabilities that govern the glycosylation network, a fitting technique was used to calibrate the model to an experimental baseline - a technique commonly used in previous kinetic models. The challenge presented by glycosylation is that the network is severely underdetermined (i.e. there are more reactions than metabolites). This means that there are often many mathematical solutions for any given glycoform. This problem was solved by using Monte Carlo sampling to create a distribution of many randomly generated solutions. These solutions can be averaged to generate a general transition probability matrix with confidence limits. Knockdown, knockout, and upregulation experiments can be simulated by modifying the transition probabilities of all reactions catalyzed by the affected enzyme(s) appropriately.

They validated their model on two separate experiments from literature. The first experiment involved a knockout of *FUT8*, the gene that codes for fucosyltransferase, in a culture of IgG-producing CHO cells. The model was able to

exactly match the wild-type baseline glycoform for the experiment. To simulate a *FUT8* knockout, the transition probability for all reactions catalyzed by fucosyltransferase were set to zero and the model was re-run. The model qualitatively predicted all glycans correctly and the deviation for each predicted glycan was under 1%, and the error for each of the three predicted glycans were 9% (biantennary glycan), 8% (biantennary, single-galactosylated glycan), and 3% (biantennary, double-galactosylated glycan). The second experiment consisted of an *MgatIV* knockout (which eliminates production of the enzyme GnTIV). After calibration, the knockout was simulated by setting all reactions catalyzed by GnTIV to zero. The results had more variability than the previous experiment, with average confidence limits of approximately 12% for the four glycans measured. The error of each glycan predicted by the model when compared with experiment was 10%, 1%, 8%, and 2%.

This model is the first to forgo kinetic representations of glycosylation in favor of a constraint-based, network approach. Though it is capable of accurately predicting glycosylation changes due to gene knockouts, there are key flaws which should be addressed in future models. First, the model cannot incorporate the co-substrate generation metabolism into the model. Second, as predicting the effects of experiments involve changing the transition probabilities for appropriate reactions, it is unlikely the user will know how to alter the transition probability for overexpression and knockdown experiments *a priori*. Lastly, it is difficult to relate the hundreds of calculated transition probabilities to real biological values such as enzyme concentration/activity.

2.3.3.2 Jimenez del Val. et al., 2016

The model created by Jimenez del Val et al. aimed to explore how protein secretory capacity (proteins secreted/unit time) affects glycosylation. It is well documented that as cell culture duration is extended, the glycosylation patterns of secreted glycoproteins gradually become less processed, resulting in an increase of high mannose glycans in the glycoform. The hypothesis established in this paper is that a decrease in cellular secretory capacity may play a role in reduced glycan processing during extended cell cultures. A glycosylation model was created that links the cell growth kinetics to glycosylation to develop a framework to study these effects. The model was applied under two different methods for reducing secretory capacity. Each method was used to replicate an experimental glycoform, and the model was used to identify the required GnTI expression level to minimize the amount of high mannose glycans in extended culture.

The model proposed two mechanisms for a decrease in glycan processing through modified secretory transport. The first is that if the incoming flow of protein is raised above the baseline, the retention time of each protein decreases. This could cause a decrease in processing completion for each protein. The second possibility is that the volume of each Golgi compartment is increasing over time, diluting the concentration of both protein and glycosyltransferase and decreasing processing efficiency. Both models shared a simple cell growth model (reliant on glucose and lactose uptake, lactose secretion, and specific mAb productivity). This model was linked (through protein productivity) to a kinetic model for glycosylation. The glycosylation model was simpler than previous kinetic models, only containing eight glycosyltransferases and 95 reactions. The Golgi was assumed to be filled with

literature-derived concentrations for each co-substrate. The kinetic parameters and concentrations for all glycosyltransferases were either taken from literature or estimated using sensitivity analysis. The transport equations used to couple both models together were modified to describe both hypotheses for intra-Golgi secretory transport. To simulate changing protein production, the transport velocity of secretory cargo is changed, and to mimic the second hypothesis, the volume for each Golgi compartment is changed.

The model was tested on mAb-producing CHO cells and was found to successfully replicate the observed growth curve. The glycoform between model and experiment was compared to assess the accuracy of the algorithm used to fit the glycosyltransferase kinetic parameters to the data. Model and experiment had good agreement, with high mannose glycans (M5) increasing by over 15% throughout the culture. The model was then used to examine optimal strategies one would employ to maintain the level of M5 glycans at starting levels. The most straightforward way to do this is increase the expression of GnTI, an enzyme essential for attaching a GlcNAc residue to a free $\alpha(1, 3)$ mannose branch of the glycan. The model found that the concentration of GnTI would need to triple to maintain the optimal levels of M5. The results for both abstractions of secretory transport were essentially the same, with the Golgi enlargement theory exhibiting a slightly delayed response in glycoform modification due across the culture.

This model is significantly simpler than previously developed kinetic models. However, it was used to study an important underlying cause for suboptimal glycoforms in extended cell culture. One weakness is assumption that enzyme

concentrations are constant for the duration of cell culture is often not applicable to many experiments.

2.4 Overview of Constraint-based Modeling

Constraint-based modeling has been shown to successfully enable quantitative analysis of large-scale genomic and metabolic networks (Bordbar et al., 2014). In contrast to kinetic models, which describe a system dynamics in great detail, constraint-based models focus on mapping the flow of metabolites through large-scale metabolic networks without the use of kinetic parameters (Orth et al., 2010). The advantage of this paradigm shift is that constraint-based models do not need detailed kinetic parameters and can easily model systems across multiple time scales.

Constraint-based modeling has gone through a number of changes over the years. It was initially developed as a means to predict the physiological consequences of acetate overflow in the *E. coli* metabolism (Majewski & Domach, 1989). The initial success was expanded upon by developing and refining different objective functions that aid in calculating accurate predictions (Savinelli et al., 1992; Schuetz et al., 2007). These models were then applied to a genome-scale representation of *E. coli* to predict growth rate from uptake rates of acetate and oxygen (Edwards et al., 2001). As transcriptomic and proteomic methods matured, the resulting data was incorporated into constraint-based models to better understand the relationships between an organism's genotype and phenotype. For example, Patil & Nielsen used constraint-based analysis to develop a model of transcriptional regulatory networks for yeast

cells (Patil et al., 2004). These efforts have validated the utility of constraint-based methods in modeling multi-scale biological systems.

Constraint-based analysis is a term that encompasses a number of algorithms with which to build and evaluate metabolic networks (Schellenburger et al., 2011). The workflow for generating a constraint-based model can be seen in Figure 2.6. First, the reaction network of n reactions and m metabolites is abstracted into a numeric matrix $[S]$ of size $m*n$. Then, a suitable constraint-based algorithm and objective function is applied to produce a flux solution for all reactions in the network. Predictions can then be made by restricting the flux from certain pathways and evaluating the network response. Flux balance analysis (FBA), one of the most common constraint-based algorithms, aims to solve Equation 2.16 for v , where $[S]$ is the stoichiometric matrix of size $m*n$ for all reactions in a system and v is a vector of flux values for all reactions (Orth et al., 2010). Assuming steady-state, this equation naturally imposes the mass balance on the reaction network. However, this may not be sufficient when describing large-scale metabolic networks with more reactions than metabolites (*i.e.* $n > m$). In this situation, the system will be underdetermined and as a result will contain multiple solutions. As FBA only selects a single viable solution from the solution space, direct application of FBA to an underdetermined system may not give useful results. To solve this problem, mathematical constraints are applied to shrink the number of viable solutions in an underspecified network. Each reaction in the network can have an upper and lower bound, which represents respectively the upper and lower limits of the reaction's calculated value when solving the flux balance problem.

$$[S]v = \frac{dv}{dt} = 0 \quad (2.16)$$

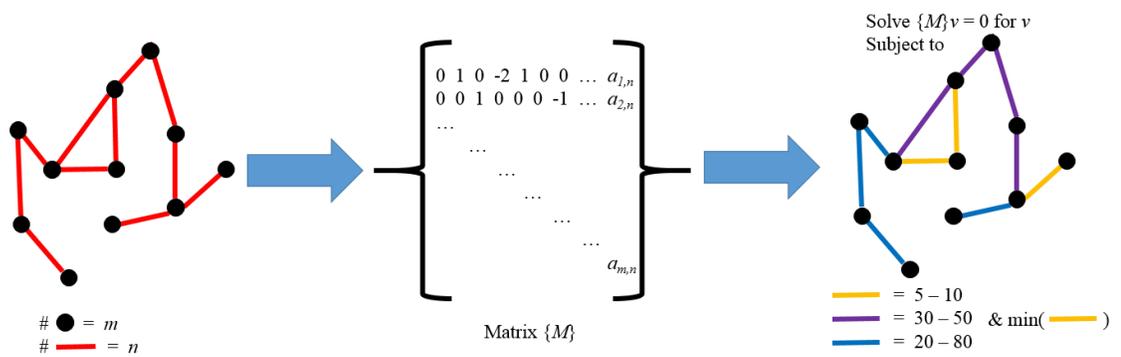


Figure 2.6: Transformation of a physiological reaction network into a constraint-based model.

There are multiple ways to impose constraints on the solution vector v . For example, if a network input such as glucose has a biologically known uptake value, constraints can be used to set the model's uptake of that component to a constant level. If there is an upper limit to the flux for a particular reaction via enzyme saturation, the reactions in question can be capped at a specific value without restricting the lower bound. Irreversible reactions have a lower bound of zero by default, and reactions catalyzed by enzymes whose genes have been knocked out exhibit an upper and lower bound of zero.

An additional type of constraint commonly used in flux-balance models is called an objective function (Segré et al., 2002). An objective function consists of a set of reactions in the network that must be maximized or minimized before an acceptable solution can be achieved. There have been many types of objective functions proposed, such as maximizing production of biomass or minimizing the Euclidian change in flux values after a specific gene knockout. Objective functions, though based on reasonable assumptions, have been used with varying degrees of success in

literature, and there is no single objective function guaranteed to maximize model performance (Schuetz et al., 2007).

If these techniques for applying constraints are insufficient for obtaining a small enough solution space, flux variability analysis (FVA) can be performed on the reaction network (Bordbar et al. 2014). The relationship between solutions obtained through FBA, an objective function, and FVA can be seen in Figure 2.7. Rather than obtaining a single solution with FBA, FVA finds the maximum and minimum value of each reaction that satisfies all of the model's constraints, including the objective function. As constraint-based analysis is a tool that can be used to quantitatively model large-scale metabolic and genomic networks (Thiele & Palsson, 2010), we have used these techniques in our construction of a low-parameter model for glycosylation.

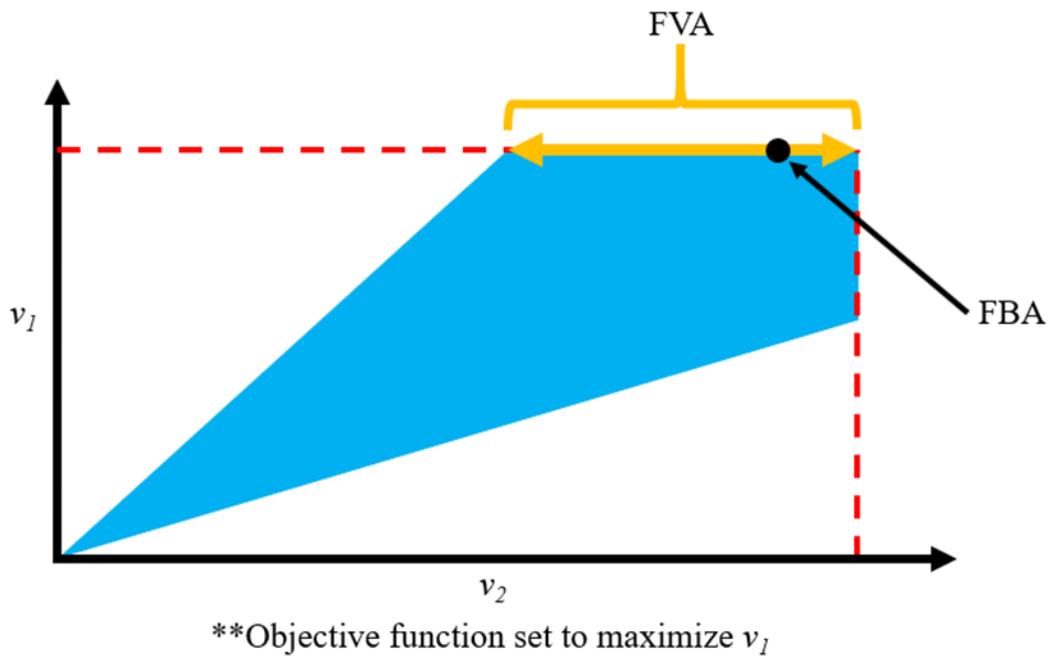


Figure 2.7: Illustration between the FBA and FVA algorithms for hypothetical solution space (shown in blue) for reactions v_1 and v_2 . Constraints within the solution space are represented as red dashed lines.

REFERENCES

- Ahn, W. S., & Antoniewicz, M. R. (2011). Metabolic flux analysis of CHO cells at growth and non-growth phases using isotopic tracers and mass spectrometry. *Metabolic engineering*, *13*(5), 598-609.
- Badur, M. G., Zhang, H., & Metallo, C. M. (2015). Enzymatic passaging of human embryonic stem cells alters central carbon metabolism and glycan abundance. *Biotechnology journal*, *10*(10), 1600-1611.
- Baker, K. N., Rendall, M. H., Hills, A. E., Hoare, M., Freedman, R. B., & James, D. C. (2001). Metabolic control of recombinant protein N-glycan processing in NS0 and CHO cells. *Biotechnology and bioengineering*, *73*(3), 188-202.
- Balch, W. E., Dunphy, W. G., Braell, W. A., & Rothman, J. E. (1984). Reconstitution of the transport of protein between successive compartments of the Golgi measured by the coupled incorporation of N-acetylglucosamine. *Cell*, *39*(2), 405-416.
- Bonfanti, L., Mironov Jr, A. A., Martínez-Menárguez, J. A., Martella, O., Fusella, A., Baldassarre, M., ... & Luini, A. (1998). Procollagen traverses the Golgi stack without leaving the lumen of cisternae: evidence for cisternal maturation. *Cell*, *95*(7), 993-1003.
- Bordbar, A., Monk, J. M., King, Z. A., & Palsson, B. O. (2014). Constraint-based models predict metabolic and associated cellular functions. *Nature Reviews Genetics*, *15*(2), 107.
- Brooks, S., Dwek, M., & Schumacher, U. (2002). *Functional and molecular glycobiology*. Bios Scientific.

Chen, P., & Harcum, S. W. (2006). Effects of elevated ammonium on glycosylation gene expression in CHO cells. *Metabolic Engineering*, 8(2), 123-132.

Edwards, J. S., Ibarra, R. U., & Palsson, B. O. (2001). In silico predictions of Escherichia coli metabolic capabilities are consistent with experimental data. *Nature biotechnology*, 19(2), 125.

Farquhar, M. G. (1985). Progress in unraveling pathways of Golgi traffic. *Annual review of cell biology*, 1(1), 447-488.

Galleguillos, S. N., Ruckerbauer, D., Gerstl, M. P., Borth, N., Hanscho, M., & Zanghellini, J. (2017). What can mathematical modelling say about CHO metabolism and protein glycosylation?. *Computational and structural biotechnology journal*, 15, 212-221.

Gawlitzeck, M., Valley, U., & Wagner, R. (1998). Ammonium ion and glucosamine dependent increases of oligosaccharide complexity in recombinant glycoproteins secreted from cultivated BHK-21 cells. *Biotechnology and bioengineering*, 57(5), 518-528.

Glick, B. S., Elston, T., & Oster, G. (1997). A cisternal maturation mechanism can explain the asymmetry of the Golgi stack. *FEBS letters*, 414(2), 177-181.

Gramer, M. J., Eckblad, J. J., Donahue, R., Brown, J., Shultz, C., Vickerman, K., ... & van Berkel, P. H. (2011). Modulation of antibody galactosylation through feeding of uridine, manganese chloride, and galactose. *Biotechnology and bioengineering*, 108(7), 1591-1602.

Grammatikos, S. I., Valley, U., Nimtz, M., Conradt, H. S., & Wagner, R. (1998). Intracellular UDP- N-Acetylhexosamine Pool Affects N-Glycan Complexity:

A Mechanism of Ammonium Action on Protein Glycosylation. *Biotechnology progress*, 14(3), 410-419.

Gu, X., & Wang, D. I. (1998). Improvement of interferon-g sialylation in Chinese hamster ovary cell culture by feeding of N-acetylmannosamine. *Biotechnology and bioengineering*, 58(6), 642-648.

Higel, F., Seidl, A., Sörgel, F., & Friess, W. (2016). N-glycosylation heterogeneity and the influence on structure, function and pharmacokinetics of monoclonal antibodies and Fc fusion proteins. *European Journal of Pharmaceutics and Biopharmaceutics*, 100, 94-100.

Hossler, P., Mulukutla, B. C., & Hu, W. S. (2007). Systems analysis of N-glycan processing in mammalian cells. *PloS one*, 2(8), e713.

Jedrzejewski, P. M., del Val, I. J., Constantinou, A., Dell, A., Haslam, S. M., Polizzi, K. M., & Kontoravdi, C. (2014). Towards controlling the glycoform: a model framework linking extracellular metabolites to antibody glycosylation. *International journal of molecular sciences*, 15(3), 4492-4522.

Jimenez del Val, I., Fan, Y., & Weilguny, D. (2016). Dynamics of immature mAb glycoform secretion during CHO cell culture: An integrated modelling framework. *Biotechnology journal*, 11(5), 610-623.

Jimenez del Val, I., Nagy, J. M., & Kontoravdi, C. (2011). A dynamic mathematical model for monoclonal antibody N-linked glycosylation and nucleotide sugar donor transport within a maturing Golgi apparatus. *Biotechnology progress*, 27(6), 1730-1743.

Kawano, S., Hashimoto, K., Miyama, T., Goto, S., & Kanehisa, M. (2005). Prediction of glycan structures from gene expression data based on glycosyltransferase reactions. *Bioinformatics*, *21*(21), 3976-3982.

Kontoravdi, C., Asprey, S. P., Pistikopoulos, E. N., & Mantalaris, A. (2007). Development of a dynamic model of monoclonal antibody production and glycosylation for product quality monitoring. *Computers & chemical engineering*, *31*(5-6), 392-400.

Krambeck, F. J., Bennun, S. V., Narang, S., Choi, S., Yarema, K. J., & Betenbaugh, M. J. (2009). A mathematical model to derive N-glycan structures and cellular enzyme activities from mass spectrometric data. *Glycobiology*, *19*(11), 1163-1175.

Krambeck, F. J., & Betenbaugh, M. J. (2005). A mathematical model of N-linked glycosylation. *Biotechnology and Bioengineering*, *92*(6), 711-728.

Majewski, R. A., & Domach, M. M. (1990). Simple constrained-optimization view of acetate overflow in E. coli. *Biotechnology and bioengineering*, *35*(7), 732-738.

McAtee, A. G., Templeton, N., & Young, J. D. (2014). Role of Chinese hamster ovary central carbon metabolism in controlling the quality of secreted biotherapeutic proteins. *Pharmaceutical Bioprocessing*, *2*(1), 63-74.

Monica, T. J., Andersen, D. C., & Goochee, C. F. (1997). A mathematical model of sialylation of N-linked oligosaccharides in the trans-Golgi network. *Glycobiology*, *7*(4), 515-521.

Opat, A. S., van Vliet, C., & Gleeson, P. A. (2001). Trafficking and localisation of resident Golgi glycosylation enzymes. *Biochimie*, *83*(8), 763-773.

Orci, L., Glick, B. S., & Rothman, J. E. (1986). A new type of coated vesicular carrier that appears not to contain clathrin: its possible role in protein transport within the Golgi stack. *Cell*, *46*(2), 171-184.

Orth, J. D., Thiele, I., & Palsson, B. Ø. (2010). What is flux balance analysis?. *Nature biotechnology*, *28*(3), 245.

Patil, K. R., Åkesson, M., & Nielsen, J. (2004). Use of genome-scale microbial models for metabolic engineering. *Current opinion in biotechnology*, *15*(1), 64-69.

Pelham, H. R., & Rothman, J. E. (2000). The debate about transport in the Golgi—two sides of the same coin?. *Cell*, *102*(6), 713-719.

Rabouille, C., Hui, N., Hunte, F., Kieckbusch, R., Berger, E. G., Warren, G., & Nilsson, T. (1995). Mapping the distribution of Golgi enzymes involved in the construction of complex oligosaccharides. *Journal of cell science*, *108*(4), 1617-1627.

Savinell, J. M., & Palsson, B. O. (1992). Optimal selection of metabolic fluxes for in vivo measurement. I. Development of mathematical methods. *Journal of theoretical biology*, *155*(2), 201-214.

Schellenberger, J., Que, R., Fleming, R. M., Thiele, I., Orth, J. D., Feist, A. M., ... & Kang, J. (2011). Quantitative prediction of cellular metabolism with constraint-based models: the COBRA Toolbox v2. 0. *Nature protocols*, *6*(9), 1290.

Schuetz, R., Kuepfer, L., & Sauer, U. (2007). Systematic evaluation of objective functions for predicting intracellular fluxes in *Escherichia coli*. *Molecular systems biology*, *3*(1), 119.

Segre, D., Vitkup, D., & Church, G. M. (2002). Analysis of optimality in natural and perturbed metabolic networks. *Proceedings of the National Academy of Sciences*, *99*(23), 15112-15117.

Senger, R. S., & Karim, M. N. (2008). Prediction of N-linked glycan branching patterns using artificial neural networks. *Mathematical biosciences*, *211*(1), 89-104.

Shelikoff, M., Sinskey, A. J., & Stephanopoulos, G. (1996). A modeling framework for the study of protein glycosylation. *Biotechnology and bioengineering*, *50*(1), 73-90.

Spahn, P. N., Hansen, A. H., Hansen, H. G., Arnsdorf, J., Kildegaard, H. F., & Lewis, N. E. (2016). A Markov chain model for N-linked protein glycosylation—towards a low-parameter tool for model-driven glycoengineering. *Metabolic engineering*, *33*, 52-66.

St. Amand, M. M., Tran, K., Radhakrishnan, D., Robinson, A. S., & Ogunnaike, B. A. (2014). Controllability analysis of protein glycosylation in CHO cells. *PloS one*, *9*(2), e87973.

St Amand, M. M., Radhakrishnan, D., Robinson, A. S., & Ogunnaike, B. A. (2014). Identification of manipulated variables for a glycosylation control strategy. *Biotechnology and bioengineering*, *111*(10), 1957-1970.

Thiele, I., & Palsson, B. Ø. (2010). A protocol for generating a high-quality genome-scale metabolic reconstruction. *Nature protocols*, *5*(1), 93.

Umaña, P., & Bailey, J. E. (1997). A mathematical model of N-linked glycoform biosynthesis. *Biotechnology and bioengineering*, *55*(6), 890-908.

Wong, N. S., Wati, L., Nissom, P. M., Feng, H. T., Lee, M. M., & Yap, M. G. (2010). An investigation of intracellular glycosylation activities in CHO cells: effects of nucleotide sugar precursor feeding. *Biotechnology and bioengineering*, *107*(2), 321-336.

Yang, M., & Butler, M. (2002). Effects of ammonia and glucosamine on the heterogeneity of erythropoietin glycoforms. *Biotechnology progress*, 18(1), 129-138.

Chapter 3

CONSTRAINT-BASED FRAMEWORK FOR GLYCOSYLATION MODELING

3.1 Introduction

Our model uses constraint-based analysis techniques to quantitatively study and predict glycosylation patterns. It was designed to require only the glycoform as an input parameter, similar to previous glycosylation models developed in the Lee lab (Kremkow, submitted). It includes all major glycosyltransferases in the Golgi (Kim et al., 2009) as well as all reactions that produce the required co-substrates (McAtee et al., 2014). It also links these reactions to the appropriate genes, giving the user a tool to predict and study glycosylation and its associated gene functions. This section will give a brief overview on the computational structure of our model, including an explanation of all functions used for glycosylation analysis.

It is important to note that as constraint-based models calculate flux solutions in a metabolic network, the units of all solutions are formally written as molecules of metabolite/unit time. However, in our model, the units are written as the number of metabolites/100 glycans produced (i.e., percentage of total glycoform). As the glycan production rate is naturally a function of time and changes across culture conditions, culture duration, and cell line, we do not include a ‘per unit time’ clause when reporting model output.

3.2 Model Structure

As previously explained, there are two major components to complete glycosylation: the presence and action of the major glycosyltransferases, and the synthesis of all sugar co-substrates. As the glycosyltransferases link a sugar to a glycan, it cannot act if the correct co-substrate is absent. Our model is designed as shown in Figure 3.1 to account for both glycosyltransferase action and co-substrate metabolism (referred to as Enzyme Analysis and Metabolism Analysis, respectively). However, because our model does not contain kinetic information, combining both mechanisms in a single, unified model would result in many confounding parameter effects. For example, without kinetic information, it is impossible to tell whether a drop in mAb fucosylation is due to a decrease in the activity/concentration of fucosyltransferase or a decrease in the GDP-Fuc co-substrate. As a result, we have kept the Enzyme Analysis and Metabolism Analysis portions of our model separate in accordance with the assumptions stated in Figure 3.1. Our model is used in Enzyme Analysis Mode if the user assumes that the experimental change is dealing with a modification in the activity/concentration of a glycosyltransferase, such as a knockout or knockdown study. If the user's experiment is changing the media supplementation (and subsequently the co-substrate metabolism) to alter glycosylation, the model can be run in Metabolism Analysis Mode to study observed changes in the co-substrate generation network. As most literature experiments manipulate either the glycosyltransferase composition or the media composition, we believe this assumption is a reasonable way to analyze the glycoform in the absence of kinetic information.

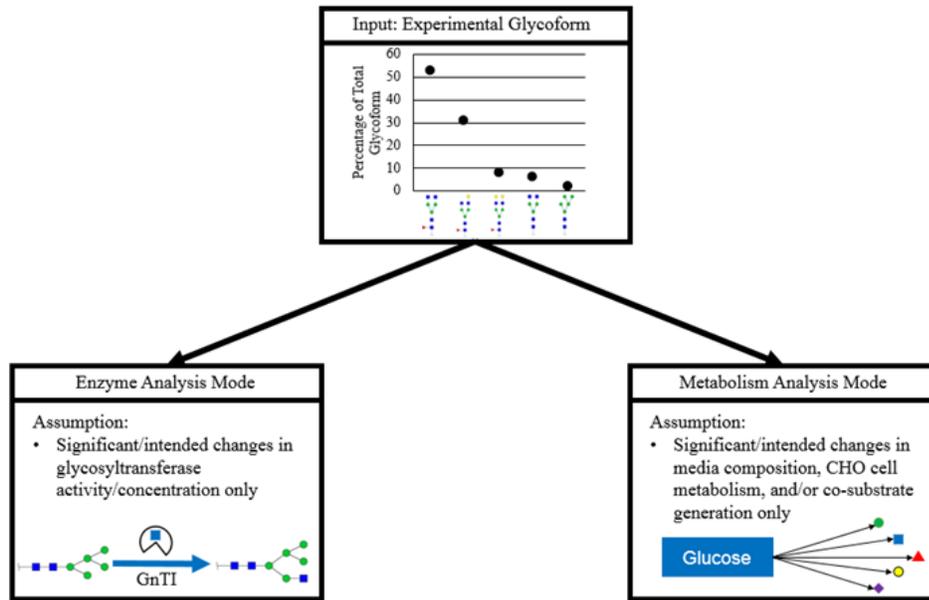


Figure 3.1: Model assumptions used in both modes of operation using glycoform as only input.

Within each mode of analysis, a number of constraint-based and custom fitting tools are included to provide quantitative analysis and prediction of glycosylation. Two platforms were used for our model: Matlab and Microsoft Excel. The relationship between these two methods of analysis can be seen in Figure 3.2. While Excel served as the database for all glycosylation reactions, genes, enzymes, and results, Matlab was used to gather this input data, create a constraint-based model, perform the necessary analysis, and return a set of flux solutions or parameters.

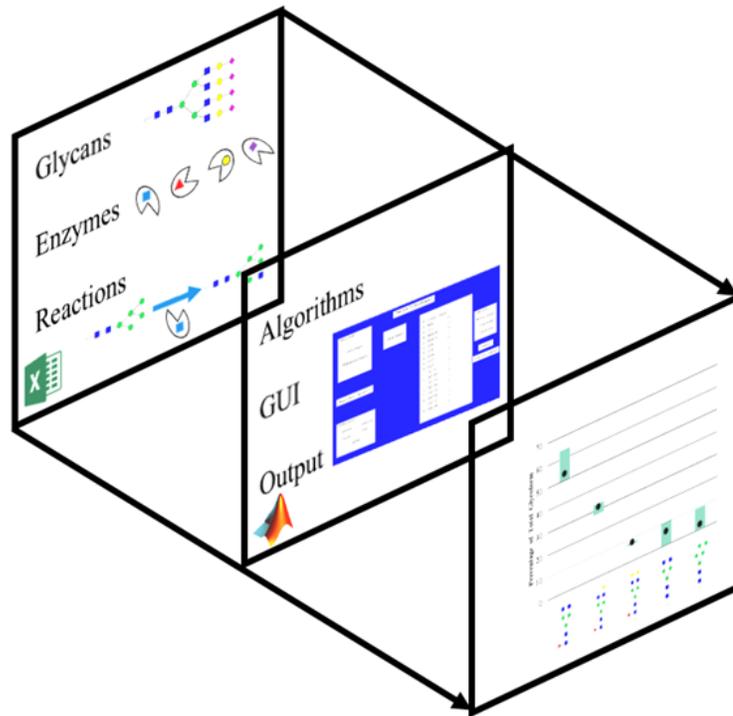


Figure 3.2: Organizational structure for programming and operating constraint-based model.

Without the use of kinetic parameters, our model has no way to modulate system dynamics through manipulation of key physical parameters such as enzyme concentration or protein flowrate, which was the method of glycosylation control adopted by previous models. To overcome this limitation, we have implemented a parameter-driven control system that serves as a means for the user to manipulate the intra-network flux distribution and control the simulated glycoform. Figure 3.3 shows how this system is implemented. For a given network, a series of unitless enzyme capacity parameters (ECPs) are defined which represent the activity/concentration of key glycosyltransferases in the network. Each ECP dictates the allowed flux of all reactions or a subset of reactions catalyzed by the respective enzyme (Figure 3.3 (A)).

In this way, a reaction network can be controlled by modification of these ECP values. Figure 3.3 (B) shows how this is implemented for a single enzyme GnTI. This enzyme catalyzes a single reaction, where a GlcNAc residue is placed on a high mannose glycan. The ECP for GnTI can be modulated to control the conversion for this reaction. As the ECP variable increases, more material is allowed through the reaction, allowing the reaction to range from completely inactive ($ECP_{GnTI} = 0$) to maximum conversion ($ECP_{GnTI} = 10$). This system can be applied to every glycosyltransferase in the N-glycosylation network to represent different glycoforms as sets of unique ECP values. Conversely, the ECP values required to produce a given glycoform can be back-calculated from the glycoform itself.

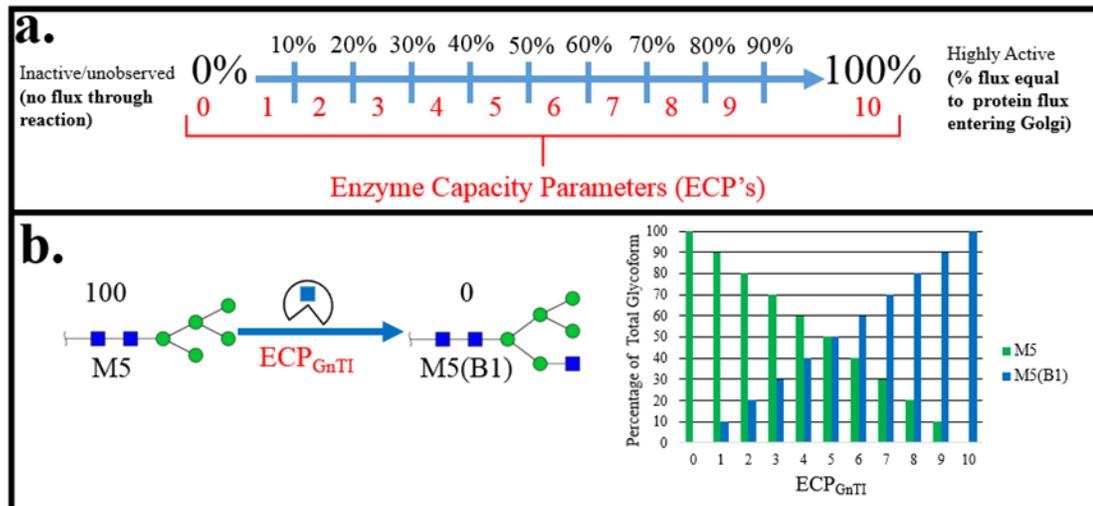


Figure 3.3: (A) Discretization of reaction flux range (normalized to incoming M9 glycan flux) into ten ECPs. (B) Application of ECP for a single reaction converting M5 to M5(B1) via the enzyme GnTI. The plot on the right shows the final glycoform as a function of the ECP controlling GnTI, assuming all glycans started as M5 and only the reaction shown is considered.

Implementation of an appropriate set of ECPs can offer several advantages for glycosylation modeling. As the number of ECPs are much smaller than the number of kinetic parameters, there are fewer assumptions being made for ECP values. Additionally, the ECPs can be calibrated to any baseline glycoform, giving our model the flexibility to study many different CHO cell lines and therapeutic proteins. Though our model cannot predict the ECP values without a baseline glycoform, kinetic glycosylation models implement a similar calibration strategy by altering the intracellular glycosyltransferase concentrations (Krambeck et al., 2005). Lastly, calibration of the ECPs to any glycoform allows the fundamental attributes of different glycoforms to be quickly and quantitatively compared. Rather than comparing multiple glycoforms measured under different experimental conditions directly, the ECPs corresponding to each glycoform can be studied to learn insights that would otherwise be difficult to ascertain.

The following chapters will demonstrate how ECPs can be created and used for both enzymatic analysis and metabolism analysis. A series of experiments were run with our model to demonstrate how the ECPs can be successfully calibrated to unique glycoforms, predict the effects of glycosyltransferase knockouts, and isolate unexpected effects due to glycosyltransferase overexpression.

3.3 Materials and Methods

The model was designed using Microsoft Excel and Matlab R2016b (MathWorks). Generating the stoichiometric reaction matrix as well as implementing the FBA and FVA algorithms were accomplished using the CONstraint-Based

Reconstruction and Analysis (COBRA) Toolbox version 2.4 (Schallenburger et al., 2011). Currently, the COBRA Toolbox version 3.0 is available, and though our work used version 2.4, future research should be done using the updated software package (Heirendt et al., 2017).

To run the COBRA Toolbox, Matlab requires the assistance of the appropriate libraries to enable reading and writing in the Systems Biology Markup Language (SBML) format. Our work used the libSBML programming library (Bornstein et al., 2008) and the SBML Toolbox version 4.1 (Keating et al., 2006). The last component required for this model is a linear programming solver. We used the Gurobi Optimizer package for this work (Gurobi Optimization Inc., 2016).

REFERENCES

Bornstein, B. J., Keating, S. M., Jouraku, A., & Hucka, M. (2008). LibSBML: an API library for SBML. *Bioinformatics*, 24(6), 880-881.

Heirendt, L., Arreckx, S., Pfau, T., Mendoza, S. N., Richelle, A., Heinken, A., ... & Magnusdottir, S. (2017). Creation and analysis of biochemical constraint-based models: the COBRA Toolbox v3. 0. *arXiv preprint arXiv:1710.04038*.

Kim, P. J., Lee, D. Y., & Jeong, H. (2009). Centralized modularity of N-linked glycosylation pathways in mammalian cells. *PloS one*, 4(10), e7317.

Keating, S. M., Bornstein, B. J., Finney, A., & Hucka, M. (2006). SBMLToolbox: an SBML toolbox for MATLAB users. *Bioinformatics*, 22(10), 1275-1277.

Krambeck, F. J., Bennun, S. V., Narang, S., Choi, S., Yarema, K. J., & Betenbaugh, M. J. (2009). A mathematical model to derive N-glycan structures and cellular enzyme activities from mass spectrometric data. *Glycobiology*, 19(11), 1163-1175.

Kremkow, B., Lee, K. (2017) (Submitted)

MATLAB Release 2016b, The MathWorks, Inc., Natick, Massachusetts, United States.

McAtee, A. G., Templeton, N., & Young, J. D. (2014). Role of Chinese hamster ovary central carbon metabolism in controlling the quality of secreted biotherapeutic proteins. *Pharmaceutical Bioprocessing*, 2(1), 63-74.

Optimization, G. The Gurobi Optimizer, 2016.

Schellenberger, J., Que, R., Fleming, R. M., Thiele, I., Orth, J. D., Feist, A. M.,
... & Kang, J. (2011). Quantitative prediction of cellular metabolism with constraint-
based models: the COBRA Toolbox v2. 0. *Nature protocols*, 6(9), 1290.

Chapter 4

ENZYMATIC ANALYSIS MODE

4.1 Introduction

This chapter outlines the formulation and execution of the portion of our model that analyzes glycosylation as a function of glycosyltransferase concentration/activity. This is accomplished by developing the complete N-glycosylation reaction network followed by establishing an adequate collection of ECPs. Each ECP serves as a means for the user to control model output by applying appropriate constraints to a subset of reactions. Algorithms are developed to generate a glycoform from a given set of ECPs as well as calibrate all ECP values to any given glycoform. This chapter begins by outlining the model notation used to describe all relevant glycans and is followed by an explanation of reaction network development, ECP establishment, and algorithm development. Lastly, this framework will be applied to four different studies in literature to demonstrate its ability to analyze and predict changes in glycosylation due to glycosyltransferase gene knockout and overexpression experiments.

4.2 Materials and Methods

4.2.1 Glycan Notation

The glycan notation developed for this model provides an intuitive way for the user to describe all N-glycans for CHO cell glycosylation. All glycans are assumed to

possess the two initial GlcNAc residues that are bound to the protein backbone, and these residues are not included in the glycan notation. The string that represents a glycan is composed of six substrings in the following order: the bisecting GlcNAc substring, fucose substring, mannose substring, branch #1 GlcNAc substring (placed in parentheses), branch #2 GlcNAc substring (placed in braces, only needed if the number of GlcNAc residues on all branches is > 2), Galactose substring, and sialic acid substring. The numbers following the mannose, branched GlcNAc, galactose, and sialic acid substrings (one number following each substring) denote the number of residues of that type on the glycan. If a residue does not exist, the corresponding substring is eliminated entirely. For example, the glycan M3(B1){B2}G3S1 contains three mannose residues, one and two GlcNAc residues on separate branches, three galactose residues, and four sialic acid residues. A visual example of this nomenclature can be shown in Figure 4.1. While other glycosylation models use a series of numeric digits to represent glycans (Krambeck & Betenbaugh, 2005), this method was created to allow an intuitive way to use and interpret the model even if the user has a limited knowledge of glycosylation.

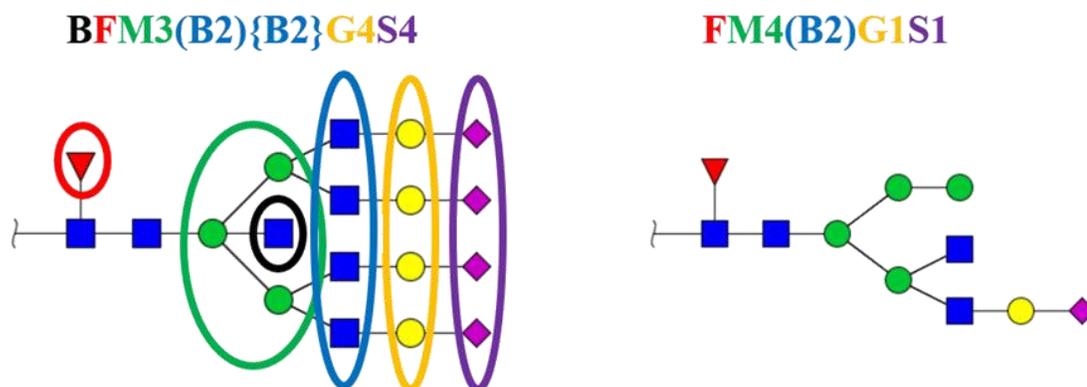


Figure 4.1: Glycan notation structure implemented in model. The colored circles correspond to the appropriately-colored substrings within the glycan string. Numbers in the string correspond to the number of appropriately-colored residues in the glycan.

4.2.2 Reaction Network Development

The central reaction network (CRN) for glycosylation in CHO cells comprises all reactions catalyzed by N-glycosyltransferases. Each glycosyltransferase has a set of substrate specifications that limit which glycans are eligible for binding. For our model, we have taken 10 glycosyltransferases from Kim et al. that represent all major modifications to glycans in CHO cells and applied the corresponding reaction rules (Table 2.1) to generate a list of all possible glycans that can be created. However, the combinatorial explosion that occurs when applying these substrate rules results in a network size on the order of thousands of glycans.

To simplify the network, we introduced a series of assumptions that was able to trim down the list of possible glycans. The first assumption is stereoisomer indistinguishability. For example, sialylation can be accomplished through two major enzymes in CHO: ST3Gal2 and ST6Gal1. These enzymes link a sialic acid residue to

a free galactose residue in an $\alpha(2, 3)$ linkage or an $\alpha(1, 6)$ linkage, respectively. Our model treats the product of both of these enzymes as the same glycan.

The second assumption deals with branch indistinguishability. As galactosyltransferase and sialyltransferase have the capability to act on any free GlcNAc or galactose residue respectively and no research has identified a therapeutic advantage for binding to any particular glycan branch, galactosylated and sialylated glycans are distinguished only by how many of each residue is on the glycan, disregarding which branch is bound to that residue.

Our last assumption deals with how our model handles glycosyltransferase localization. Research has shown that all glycosyltransferase families are localized in a relatively sequential manner in the Golgi (Rabouille et al., 1995; Dunphy et al., 1985; Houghton & Gleeson, 2001). The principle of enzyme localization in the Golgi has been described in other glycosylation models through representation of the Golgi as a series of CSTRs, each CSTR being populated with a different distribution of glycosyltransferases (Chapter 2.3). We represent this in our network topology by artificially ordering the glycosyltransferases according to an approximation of their physical localization *in vivo*. Any glycosyltransferases that exist in the same Golgi compartment are ordered according to the needs of the reaction rules. For example, galactosyltransferase and sialyltransferase are both known to inhabit the TGN. Since sialyltransferase requires a free galactose residue on a terminal glycan branch, our model has galactosyltransferase localized before sialyltransferase. These assumptions serve to simplify the reaction network by limiting the number of reactions each glycosyltransferase can facilitate while maintaining the ability to fully describe N-

glycan heterogeneity. The final reaction network consists of 278 glycans and 542 reactions. A simplified depiction of this network can be seen in Figure 4.2.

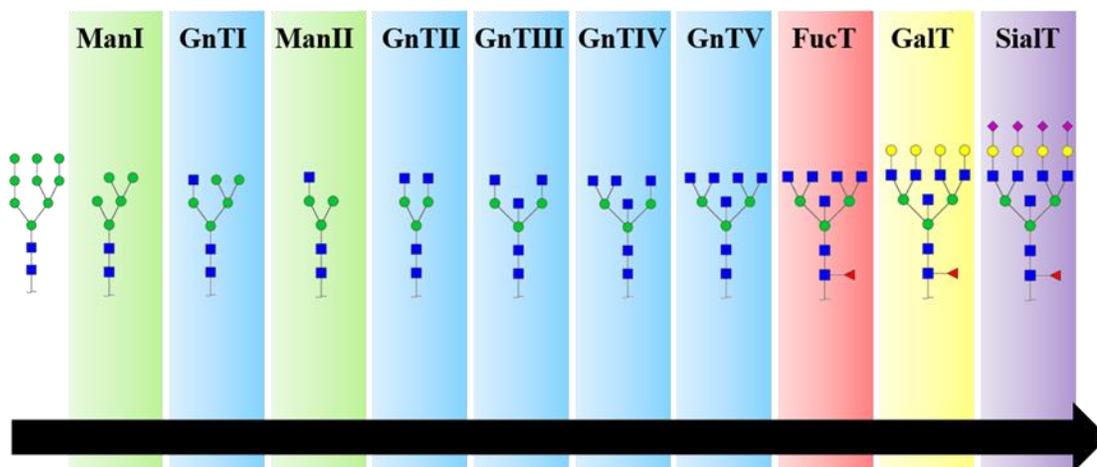


Figure 4.2: Sequence of glycosyltransferase action implemented by our model.

4.2.3 Implementation of Parameter-driven Glycosylation Modeling

The steady-state production of glycosylated proteins in CHO cells can be represented by a network of fluxes (Spahn et al., 2015). This network is analyzed with constraint-based analysis as described in Chapter 3. For this model, we employ FBA and FVA techniques to predict steady-state CHO cell glycoforms based on a simple set of parameters without the need for kinetic information. Our model takes in a steady stream of unprocessed M9 glycans and uses the ECPs to transform them into any N-glycoform observed in CHO cells based on the distribution of ECPs. The model can follow two computational algorithms. The first algorithm, called Exploration Mode, allows the user to set custom values for all ECPs and the model will return the

resulting glycoform. Alternatively, Mapping Mode calculates the set(s) of ECPs that most closely represent a glycoform entered by the user. These modes are used together to analyze and predict glycosylation experiments in literature.

Our model does not describe the actions of a single cell, rather it describes the observed glycoform for a batch of cells with constraint-based analysis. Due to this distinction, our model does not require specific protein production rates, Golgi compartment volume, enzyme concentrations, or transport coefficients to function. However, the disadvantage of this is that the model does not predict how much of each glycan is being secreted in relevant units. This means that the input flux of the starting M9 glycan is arbitrarily set to 100 glycans/unit time, and each predicted glycan can be thought of as a percent abundance rather than a concentration.

Each glycan has the opportunity to transition to any eligible glycan according to the reaction rules defined in the previous section. Each glycan also has an artificial secretion reaction that allows it to exit the network; this is representative of the glycan exiting the Golgi with no further glycosyltransferase interactions. With both transition and secretion reaction types defined, the system of equations can be solved for any distribution of N-glycans using the mass balance as the baseline constraint. This is accomplished by setting the secretion flux of each glycan to be equal to the desired normalized percentage and solving the resulting system of equations and constraints with flux-balance analysis. For example, if an observed glycoform is 40% M5(B1) and 60% M5(B1)G1, then setting the secretion flux of these two glycans to be 40 and 60 respectively, while keeping all other glycan secretions reactions set to zero, will force the network to convert the 100 incoming M9 glycans into the appropriate products.

As previously explained, constraint-based analysis methods are used for all calculations. FBA, for example, can match any glycoform by setting the constraints for all secreted glycan reactions to match the desired glycoform. However, this is not particularly useful because one cannot compare enzyme activities/concentrations nor can one modify the model to predict novel glycoforms as a result of knockdowns or knockouts. To solve these problems, our parameter system alters the reaction constraints in the model to simulate changing glycosyltransferase activities/concentrations. As explained in Chapter 3, these ECPs are discrete, unitless values that rank the combined activity and concentration of each enzyme on a scale from 0-10. The values are linearly mapped to upper constraints on all reactions catalyzed by each glycosyltransferase. A value of zero represents a glycosyltransferase with completely eliminated functionality, and all appropriate fluxes are set to zero. Conversely, a value of 10 indicates an enzyme with maximum possible conversion and all upper-flux constraints are set to equal the flux of incoming M9 glycans for reactions catalyzed by that enzyme. One can then use flux variability analysis to obtain a range, rather than a value, for each glycan that quantitatively describes its predicted percentage abundance in the glycoform. However, simply changing a reaction's upper bound often does not limit the solution space sufficiently such that a reasonable glycoform can be generated. Therefore, we have imposed lower bounds (*LB*) to all reactions that are equivalent to 90% of the corresponding upper bound (*UB*), as shown in Equation 4.1. A list of all ECPs used in the model are shown in Table 4.1.

$$LB = (0.9)UB \quad (4.1)$$

Table 4.1: List of ECP values and corresponding glycosyltransferase (or glycosyltransferase branch, terminated with “_B*”).

#	ECP	Enzyme
1	ManI	Mannosidase-I
2	GnTI	N - acetylglucosaminyltransferase-I
3	ManII_B1	Mannosidase-II ($\alpha(1,3)$ -branch)
4	ManII_B2	Mannosidase-II ($\alpha(1,6)$ -branch)
5	GnTII	N - acetylglucosaminyltransferase-II
6	GnTIII	N - acetylglucosaminyltransferase-III
7	GnTIV	N - acetylglucosaminyltransferase-IV
8	GnTV	N - acetylglucosaminyltransferase-V
9	FucT	Fucosyltransferase
10	GalT_B1	Galactosyltransferase ($\beta(1,2)$ -branch #1)
11	GalT_B2	Galactosyltransferase ($\beta(1,2)$ -branch #2)
12	GalT_B3	Galactosyltransferase ($\beta(1,4)$ -branch)
13	GalT_B4	Galactosyltransferase ($\beta(1,6)$ -branch)
14	SialT_B1	Sialyltransferase ($\beta(1,2)$ -branch #1)
15	SialT_B2	Sialyltransferase ($\beta(1,2)$ -branch #2)
16	SialT_B3	Sialyltransferase ($\beta(1,4)$ -branch)
17	SialT_B4	Sialyltransferase ($\beta(1,6)$ -branch)

As shown in Table 4.1, a total of 17 ECPs were chosen to model the glycosyltransferase-mediated reaction network. In general, we attempted to use one ECP per glycosyltransferase. However, during initial testing, we found that glycosyltransferases with the capability of binding to more than one branch on a glycan exhibit different kinetic behavior, and one ECP per glycosyltransferase was insufficient to adequately describe most glycoforms. Therefore, we added new ECPs for glycosyltransferases that can act on multiple glycan branches. This resulted in two parameters for ManII and four parameters for GalT and SialT.

4.2.4 Fitting Algorithm Development

A total of 17 parameters are used to describe the behavior of 10 glycosyltransferases commonly observed in CHO cell glycosylation. Though this is significantly less than the number of parameters used in most glycosylation models, attempting to calibrate these parameters to a glycoform by hand is tedious and requires a thorough knowledge of glycosylation. There are two algorithms built into the model that are designed to give the user reasonable estimates for all ECP values given a starting glycoform. Both algorithms work with different assumptions, and the user has the option of selecting either algorithm to calibrate model parameters to a glycoform.

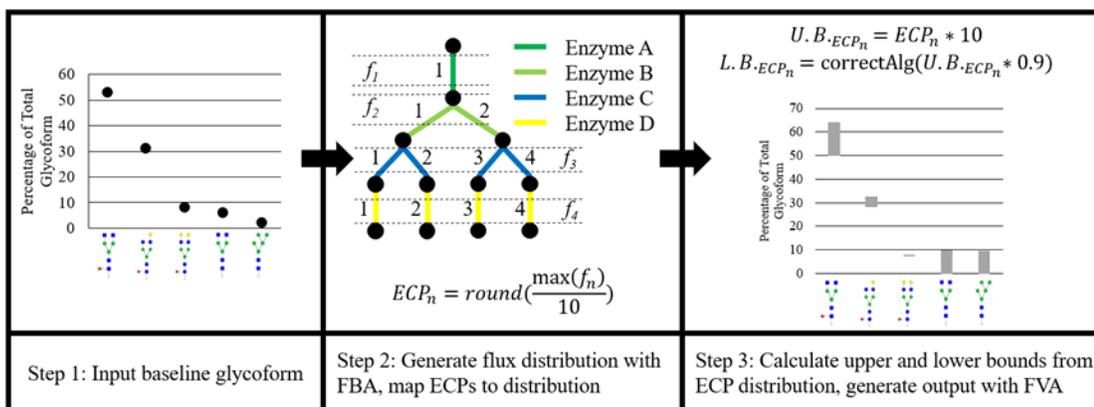


Figure 4.3: Workflow for Alg1. In Step 2, the *round()* function rounds the value in the parenthesis to the nearest integer. In Step 3, the *correctAlg()* function determines if the flux value inside the parenthesis exceeds the solution obtained via FBA in step 2. If so, it lowers the value in the parentheses to match this value.

The workflow for the first algorithm (Alg1) can be seen in Figure 4.3. Alg1 solves an unconstrained FBA problem to match the query glycoform. Then, it examines the resulting flux solution and records the highest flux value for each ECPs respective set of reactions. This will return a list of 17 flux values between 0 and 100. Each ECP is then calculated by dividing the flux value by 10 and rounding to the nearest whole number. The upper and lower bounds for the constraint-based model are calculated as shown in Figure 4.3. In these equations, the upper bound is a simple mapping of each ECP to a normalized flux range of 0-100 (Chapter 3). The lower bound is run through a small algorithm to ensure that the mass balance is kept. Ideally, the lower bound for all ECPs are set to be 90% of the corresponding upper bound. However, if the FBA solution (Step 2 of Figure 4.3) calculates a maximum flux value below this lower bound, the reaction's lower bound is reduced to be equal to the FBA solution. Finally, FVA is run with these bounds to calculate the maximum and minimum range for all glycans. This method is the simplest way of finding an initial set of ECPs for a glycoform, as it attempts to match all parameters to ensure their allowable flux range lies as close to the largest reaction as possible. Completion time is on the order of seconds, making Alg1 the faster of the two algorithms. However, for large glycoforms with many active enzymes, Alg1 may not give the best results. This is because there is no error metric Alg1 attempts to minimize, it simply matches the ECPs to the respective highest flux values calculated with FBA and runs the corresponding FVA problem.

To create a more comprehensive fitting algorithm that attempts to minimize error between model and the query glycoform, a second algorithm (Alg2) was developed. The workflow for this script is shown in Figure 4.4. This algorithm

exploits the fact that all glycosyltransferase modifications on glycans in our model are both distinguishable from each other and permanent. This means that the glycoform can be represented as a set of attributes our model can iteratively match through changing the ECPs. These attributes must be carefully chosen such that each attribute is affected by changing only a single ECP. A list of all attributes and their corresponding parameters are shown in Table 4.2. After using the query glycoform to identify the target values for all 17 attributes, the script begins the calibration process. This iterative procedure starts by setting all ECPs to zero and calculating the initial cumulative error E_0 , shown in Figure 4.4. Then, it increases the value of the first ECP by one and recalculates the error, storing it as E_1 . If $E_1 < E_0$, it saves the ECP value and repeats. If $E_1 > E_0$, the model reverts to the ECPs previous value and moves on to the next ECP. This repeats until all ECPs have been modified such that the model's E_0 is minimized.

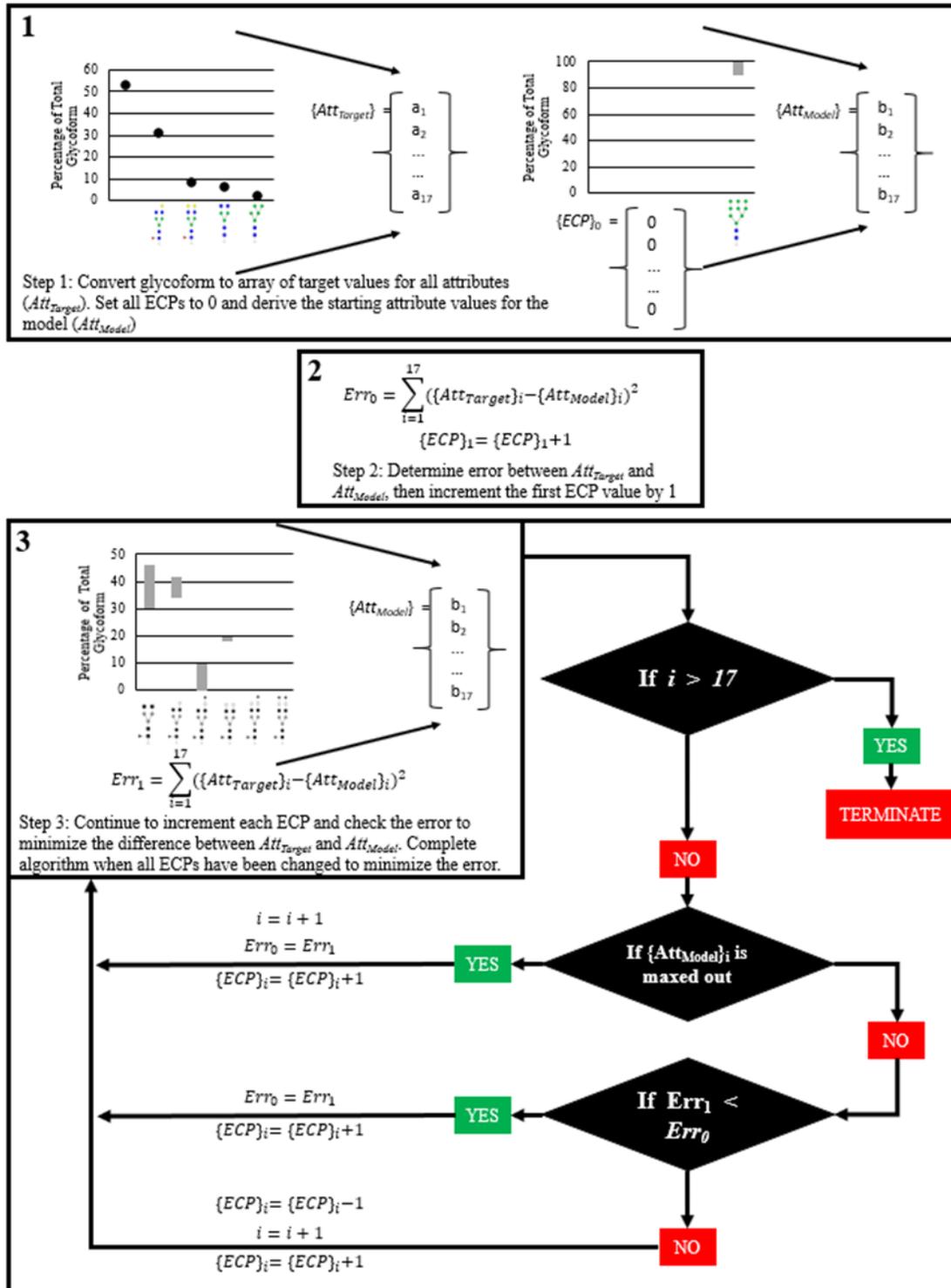


Figure 4.4: Workflow for Alg2 parameter calibration.

Table 4.2: List of all attributes used in calibration for Alg2 and the ECPs used to control each attribute.

#	Attribute	Corresponding ECP
1	% of glycans with 5, 4, or 3 mannose residues	ManI
2	% of glycans with a $\beta(1,2)$ -linked GlcNAc residue on $\alpha(1,3)$ -linked mannose	GnTI
3	% of glycans with 4 mannose residues	ManII B1
4	% of glycans with 3 mannose residues	ManII B2
5	% of glycans with a $\beta(1,2)$ -linked GlcNAc residue on $\alpha(1,6)$ -linked mannose	GnTII
6	% of glycans with bisecting GlcNAc residue	GnTIII
7	% of glycans with a $\beta(1,4)$ -linked GlcNAc residue on $\alpha(1,3)$ -linked mannose	GnTIV
8	% of glycans with a $\beta(1,6)$ -linked GlcNAc residue on $\alpha(1,6)$ -linked mannose	GnTV
9	% of fucosylated glycans	FucT
10	% of glycans with one galactose residue	GalT B1
11	% of glycans with two galactose residues	GalT B2
12	% of glycans with three galactose residues	GalT B3
13	% of glycans with four galactose residues	GalT B4
14	% of glycans with one sialic acid residue	SialT B1
15	% of glycans with two sialic acid residues	SialT B2
16	% of glycans with three sialic acid residues	SialT B3
17	% of glycans with four sialic acid residues	SialT B4

4.3 Results and Discussion

To validate our model, we have used it to accurately predict the effect(s) of simple glycosyltransferase knockouts and analyze unexpected effects of glycosyltransferase overexpression in four literature studies. This is accomplished by first calibrating our model's ECPs to reproduce the reported wild-type glycoform. A knockout is simulated by setting the appropriate ECP to zero and re-running the

model. For a glycosyltransferase overexpression (or knockdown), the corresponding ECP cannot be determined due to a lack of *a priori* information. As such, two sets of ECPs that were calibrated from both the wild-type and experimental glycoforms are compared to determine if the expression/activity of the correct enzyme was modified. Additionally, our model often generates predictions (0-10%) of intermediate glycans that are not present in the experiment. These glycans were not included in any of the following figures, but can be viewed in Appendix A. Through application of our model to predict and validate experimental results, we will show how our model can generate observations concerning changes in glycosylation that are difficult to ascertain via manual data analysis.

4.3.1 Sealover et al., 2013

The goal of this paper was to engineer CHO cells to produce a glycoform consisting primarily of high mannose glycans. This was accomplished via a complete knockout of the gene *Mgat1*, which codes for the glycosyltransferase GnT1, through a zinc-finger nuclease (ZFN) assay. Sealover et al. used a CHO-ZN glutamine synthase *-/-* cell line which produced a recombinant human IgG protein. The ZFN was designed from a homologous *Mgat1* gene in mouse. The glycoform of both the wild-type and knockout cell lines were quantified via MALDI/TOF mass spectrometry.

We first needed to determine the most accurate fitting algorithm with which to calibrate our wild-type parameter set. As explained in the Materials and Methods section, we have created two algorithms to fit our model's parameters (Alg1 and Alg2). For this glycoform, both algorithms yielded the same ECP values for the wild-

type data set. Figure 4.5 shows the fitted parameter values, the projected glycoform, and the wild-type glycoform from literature.

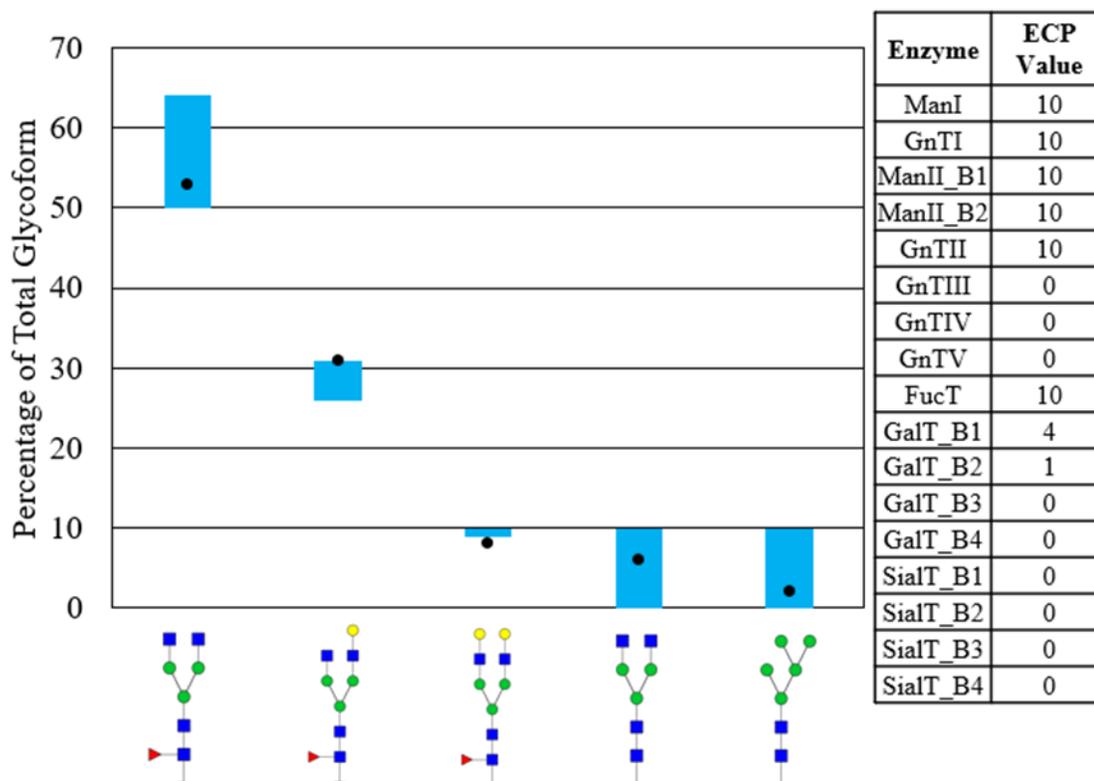


Figure 4.5: Wild-type baseline glycoform (black dots), model glycoform prediction ranges (blue bars) and corresponding ECP values (table).

The wild-type glycoform is a distribution of biantennary glycans consistent with documented glycosylation patterns of IgG proteins (Kim et al., 2010). The model's predicted ranges are consistent with the values of all experimentally-observed glycans. The majority of glycans (98%) are complex type, suggesting high conversion rates of reactions mediated by ManI, ManII, GnTI, and GnTII. The complex type

glycans are solely biantennary. This is believed to be due to the steric hindrance of the IgG protein, which inhibits the action of GnTIV and GnTV (Wormald et al., 1997). Approximately 92% of the complex glycans are fucosylated, suggesting a high conversion rate of reactions mediated by fucosyltransferase. Lastly, <20% of the complex type glycans are galactosylated, which suggests a lower wild-type conversion rate for galactosyltransferase.

The experimental glycoform data was within our model's predicted ranges for 4/5 glycans, with the exception being the FM3(B1){B1}G2 glycan. However, this glycan is only out of range by 1%. The parameter values for ManI, GnTI, ManII, GnTII, and FucT are all at their maximum value, suggesting a high activity/concentration of those enzymes relative to the flow of protein. The galactosyltransferase parameters are significantly lower, with the parameter for the first galactose (GalT_B1) branch equal to four and the second galactose branch (GalT_B2) equal to one. Additionally, the GalT_B1 parameter is four times higher than the GalT_B2 parameter. This is consistent with kinetic data on galactosyltransferase, which reports variations in terminal GlcNAc binding affinity for galactosyltransferase (Pacquet et al., 1984).

After the wild-type glycoform fitting, the model was used to predict glycosylation changes via an *Mgat1* knockout. This was accomplished by reducing the GnTI parameter to zero, while keeping all other ECPs constant. Upon knockout, the CHO cells produced a completely homogenous glycoform, with only a single high mannose type (M5) glycan remaining (Figure 4.6). This is because GnTI is a glycosyltransferase that converts high mannose type glycans to complex type glycans via the linkage of a GlcNAc residue on the free $\alpha(1, 3)$ -linked mannose branch. No

subsequent glycosyltransferases can act on a glycan unless it possesses this particular GlcNAc residue (Kim et al., 2009). For this reason, eliminating the functionality of GnTI eliminates any possibility that the M5 glycan can be a suitable substrate for other glycosyltransferases.

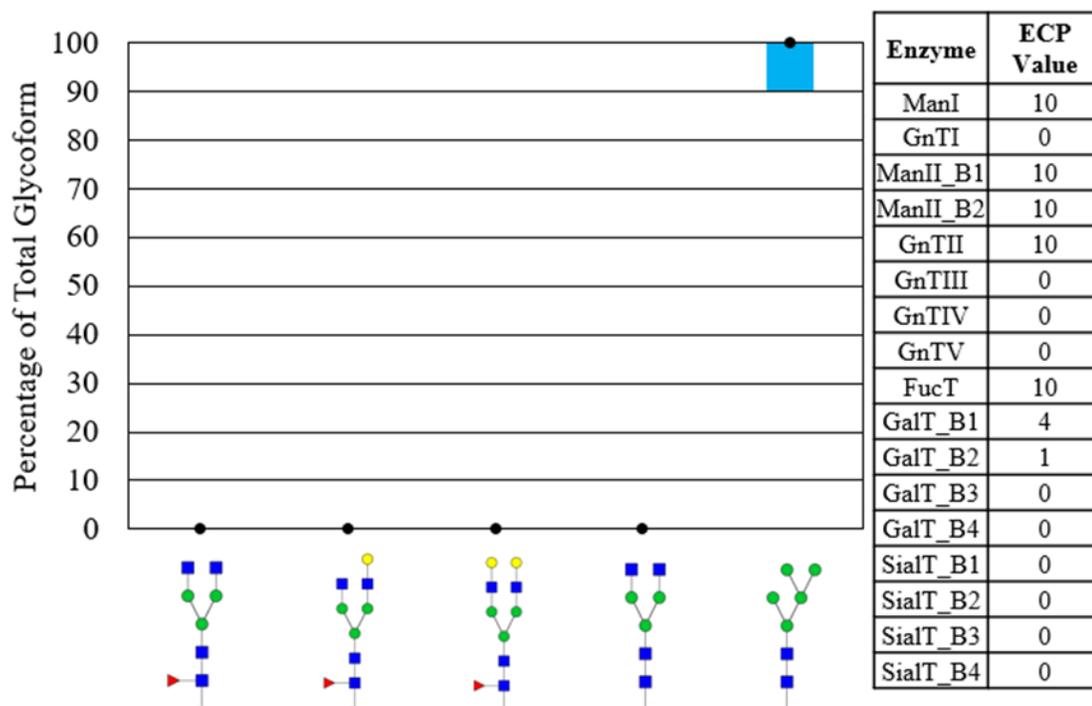


Figure 4.6: Experimental glycoform (black dots), model prediction (blue bar), and corresponding ECP values (Table) after knockout of *Mgat1* (gene that codes for production of GnTI).

Our model recapitulates this result with a complete elimination of all complex type glycans and a prediction range for M5 of 95-100%. Additionally, because only the ECP for GnTI was changed, the model still had non-zero ECP values for subsequent glycosyltransferases such as GnTII, FucT, and GalT. However, our model

correctly does not allow the M5 glycan to react with any other glycosyltransferases, validating that our model has the same network properties observed *in vivo*.

Our model can be calibrated to a standard IgG glycoform with high accuracy (4/5 glycans correctly predicted, <1% error for the incorrectly predicted glycan). Additionally, we simulated a simple knockout of GnTI successfully; our model was able to correctly predict the quantitative effects of this knockout on the glycoform.

4.3.2 Goh et al., 2014

To determine if our model can predict *Mgat1* deficiencies in more complex glycoforms, we tested our model on data presented by Goh et al., which reported a mutant CHO cell line with a disrupted *Mgat1* gene. For this experiment, CHO-K1 cells producing the glycoprotein erythropoietin were used. MALDI-TOF mass spectrometry was used to quantify the glycoforms. To estimate the resulting glycoform, we assumed that the abundance of each glycan was proportional to peak intensity of the mass spectrometry data.

As erythropoietin has no spatial restrictions that limit glycosylation like IgG proteins, the resulting glycoform can be much more complex (Figure 4.7). The majority of the glycoform consists of complex glycans, with a fairly even distribution of bi, tri, and tetraantennary glycans. Unlike IgG proteins, there is no single glycan that represents the majority of the glycoform, rather the increased maximum antennarity of EPO glycans results in a highly heterogeneous glycoform.

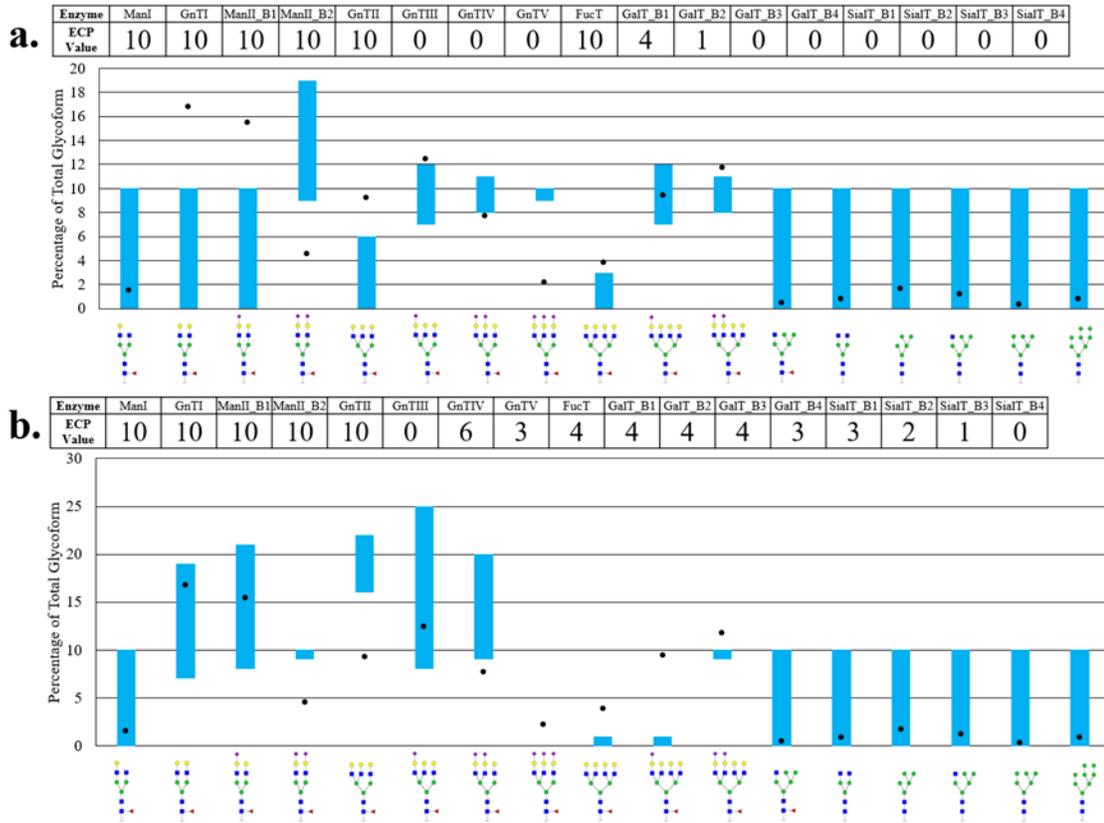


Figure 4.7: Experimental baseline glycoform values (black dots), model prediction ranges (blue bars), and ECP values (Tables) for a) Alg1 calibration b) Alg2 calibration.

We first used both fitting algorithms to get two unique estimations of the parameters that fit the baseline CHO-K1 cell line. Unlike the previous experiment, both algorithms result in unique estimations for the ECPs. Figure 4.7 shows these estimations as well as the glycoforms obtained when using both parameter sets. Alg1 returned a glycoform that contained 9 incorrectly predicted glycans out of the 17 total glycans. Alg2 predicted 7 glycans incorrectly, making Alg2 a better fit for the parameters. Closer examination of the results yields additional support for using Alg2 in this experiment. In the absence of experimental data, Alg1 gives a much less

accurate representation of the possible glycoform. This is because Alg1 predicts the top three most abundant glycans (FM3(B1){B1}G2, FM3(B1){B1}G2S1, and FM3(B2){B1}G3S1) incorrectly, while Alg2's predicted ranges encompass the experimental values for all three glycans. A reason for Alg2's superiority lies in the assumptions used to create the algorithm. While Alg1 matches the ECP according to the highest flux value in the reaction network, Alg2 calibrates the parameters according to specific characteristics of the resulting glycoform, as explained in the Materials and Methods section. For large, complex glycoforms, Alg2 uses a more conservative estimate for late-stage glycosyltransferases (GalT_B4, SialT_B1, and SialT_B2). However, the downside to Alg2 is that it fails to predict SialT_B3 as non-zero, despite there being a triple-sialylated glycan in the experimental glycoform. However, because this glycan has an abundance of only 2%, increasing this parameter from zero to one would take away from upstream glycans and reduce total accuracy.

Predicting the effects of a disrupted GnTI enzyme was accomplished by reducing the GnTI parameter to zero and re-running the model. The percentage of high mannose glycans for both model and experiment had 100% agreement. However, a look at the specific glycoform results showed differences (Figure 4.8). A number of high mannose glycan variants were visible in the experimental MS spectra for the mutant cell line which do not exist in our constructed reaction network, such as M3, M4, FM4, and FM5. These glycans are not believed to be observed due to the reaction rule constraints imposed by the glycosylation network according to literature (Kim et al., 2009; Shah et al., 2008). These glycans have not been commonly observed in other works in the field, to our knowledge. However, an alternate biochemical pathway was discovered in mouse that converts M5 glycans directly to M4 and M3 glycans without

requiring a terminal GlcNAc residue (Herscovics, 1999). This enzyme, referred to as mannosidase-III, catalyzes this pathway which was observed only after eliminating mannosidase-II functionality, suggesting its activity in healthy mammalian cells is low (Chui et al., 1997).

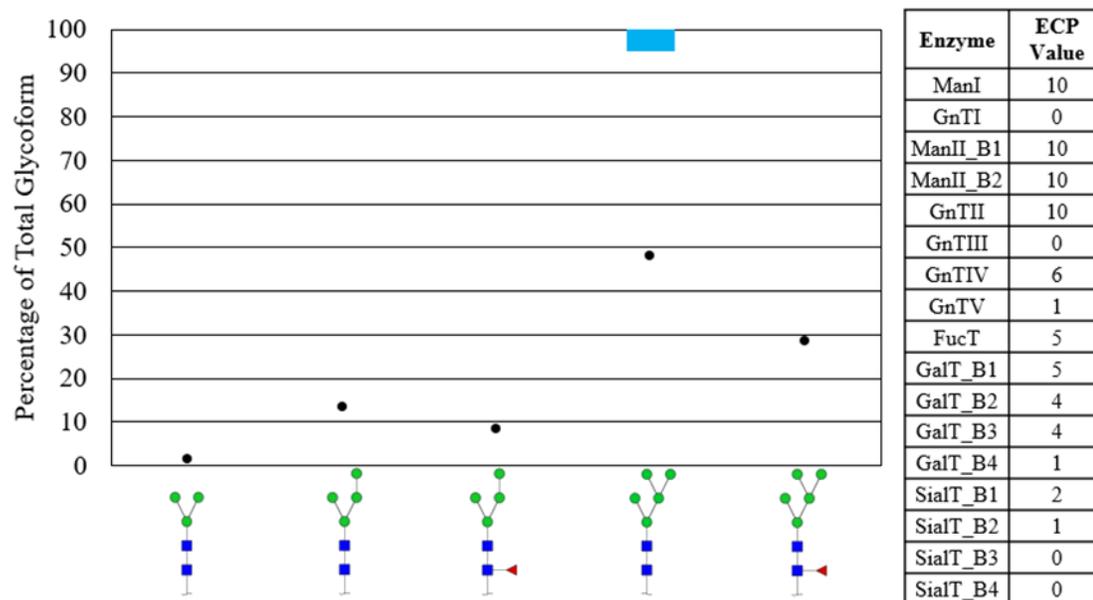


Figure 4.8: Experimental glycoform values (black dots), model prediction ranges (blue bars), and ECP values (Tables) after *MGat1* disruption.

This experiment demonstrates the ability of our model to fit complex, highly heterogeneous glycoforms with reasonable accuracy (10/17 glycans correctly predicted). Through this experiment, Alg2 was determined to be more closely representative of the glycoform than Alg1. Alg2 correctly predicted the ranges for the top three most abundant glycans, and reduction of GnTI to zero correctly eliminated all complex glycans from the network.

4.3.3 Malphettes et al., 2010

The elimination of GnTI functionality is a relatively simple modification as this glycosyltransferase controls only a single reaction in the network. To validate that this model can make reasonable predictions for knockouts that have a larger impact on the reaction network, we used our model to predict the effect(s) of a *FUT8* knockout (Malphettes et al., 2010), the gene that codes for production of fucosyltransferase (FucT). The knockout in CHO cells secreting IgG proteins was performed using zinc-finger nuclease (ZFN) techniques. The ZFN nuclease was designed according to the *FUT8* sequence obtained through sequencing of CHO cDNA. Cell growth was not appreciably altered during the knockout process, and the complete glycoforms of the wild-type and knockout CHO cell lines (4 clonal cell lines) were reported. The glycoform for all CHO knockout clones was averaged and used as the input to our model, as clonal differences in the glycoform were <8% for each glycan.

When calibrating our model to the wild-type glycoform, both algorithms had equal estimates for all ECPs (Figure 4.9). Model output matched the quantitative values for 6/7 experimentally-observed glycans. The only missed glycan, FM3(B1){B1}G2, was out of range by 2%. The wild-type glycoform consisted of mostly biantennary glycans (>94%) with a small presence of high mannose glycans (<6%). Analysis of the resulting ECPs showed a maximum value for all parameters except GalT_B1 and GalT_B2, suggesting that the activity/concentration of this enzyme is lower than that of the others. Lastly, the parameter for galactosylation of the

second branch is, again, significantly lower than galactosylation of the first branch, similar to the glycoform obtained in Chapter 4.3.1.

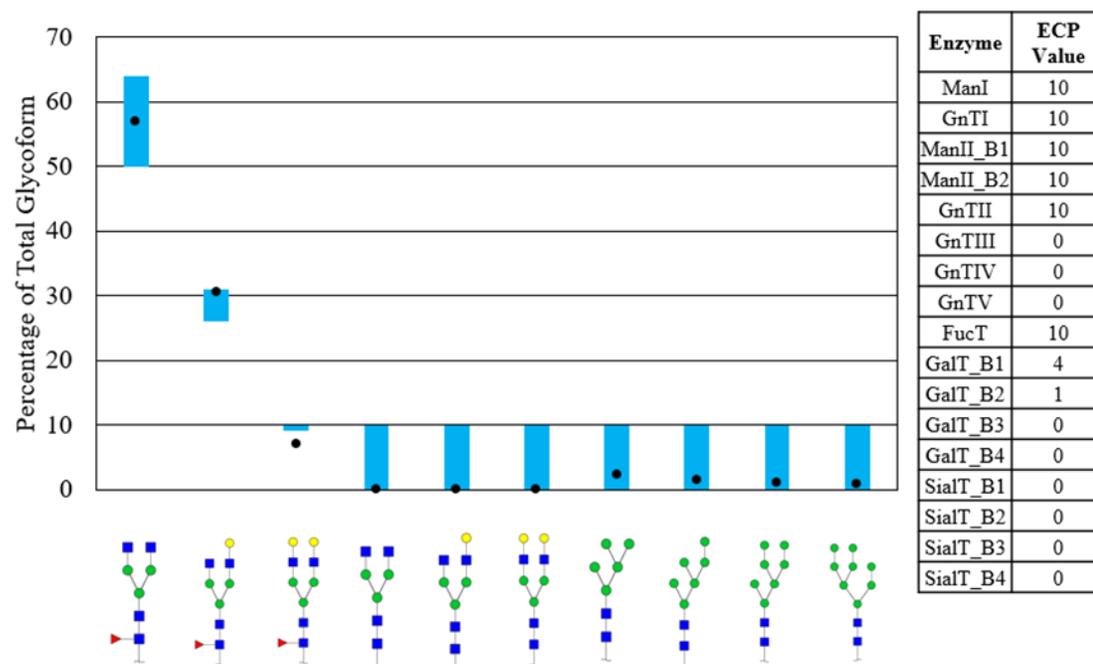


Figure 4.9: Experimental glycoform values (black dots), model prediction ranges (blue bars), and ECP values (Tables) for baseline IgG glycoform.

A knockout of *FUT8* results in a loss of FucT production, which was simulated by reducing the FucT ECP to 0 and re-running the model. As seen in Figure 4.10, the model predicted 5/7 glycans correctly, with an observed overestimate of both M3(B1){B1}G1 and M3(B1){B1}G2 by 10.4% and 3.2% respectively. This suggests that the activity/concentration of galactosyltransferase (GalT) was lowered in *FUT8* knockout cell lines. Theoretically, knocking out *FUT8* should not affect the activity/concentration of GalT. Additionally, because cell growth and viability was not

appreciably altered, it is unlikely a change in protein production rate resulted in this effect. However, there is evidence that the concentration of glycosyltransferases are highly variable. St. Amand et al. performed a series of sugar-feeding experiments on CHO cells and found that the mRNA expression level of the major N-glycosyltransferases changed when grown under different conditions (St. Amand et al., 2014). It is possible the ZFN-knockout of *FUT8* changed the endogenous expression level of galactosyltransferase. Unfortunately, there was no data included in the study to validate this hypothesis.

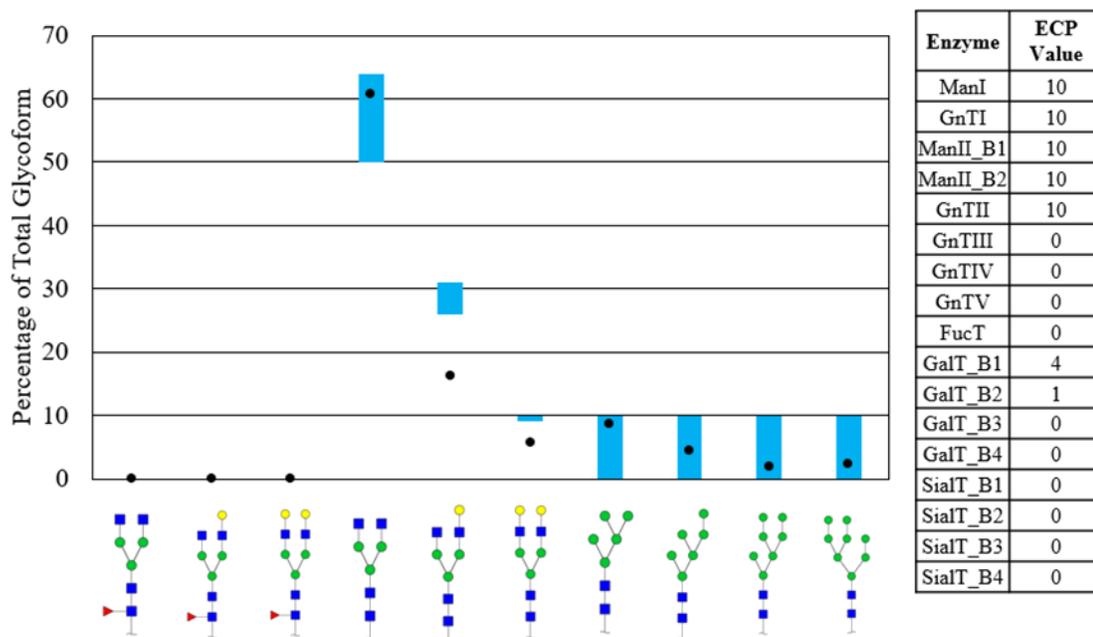


Figure 4.10: Experimental glycoform values (black dots), model prediction ranges (blue bars), and ECP values (Tables) for CHO cells with *FUT8* knockout.

The model simulation correctly predicted the top three most abundant glycans in the correct order. However, the error associated with M3(B1){B1}G1 and

M3(B1){B1}G2 suggests a change in GalT activity which was identified by our ECP system. Though this hypothesis could not be validated with the data available, this experiment shows how our model can accurately predict the knockout of FucT and identify unexpected possible flux changes in the reaction network.

4.3.4 Onitsuka et al., 2011

The previous studies involved eliminating the production of key glycosyltransferases such as GnTI and FucT through knockout experiments. However, it is also of interest to artificially increase the activity/concentration of glycosyltransferases such as sialyltransferase to improve safety and efficacy (Ghaderi et al., 2012). Onitsuka et al. increased the percentage of sialylated glycans in CHO cells through an overexpression of $\alpha(2, 6)$ sialyltransferase. Though the sequence for the gene *ST6Gall* is present in CHO cells, evidence suggests that it is not endogenously expressed (Lee et al., 1989; Xu et al., 2011). This experiment aimed to create a stable CHO-K1 cell line actively expressing *ST6Gall* and observe the change in glycosylation.

CHO-K1 cells producing IgG proteins were used for this study. The *ST6Gall* sequence was obtained through cDNA cloning from the CHO-Top-H line (Kim et al., 2010). *E. coli* was used to express and validate the correct gene sequence. The vector was amplified with PCR and transfected in CHO-K1 cells adapted to serum-free media. Methotrexate (MTX)-mediated transfection was applied, and cells were cloned with limiting dilution for two weeks to obtain the clones expressing the highest levels of *ST6Gall*. All glycoforms were analyzed with MALDO-TOF mass spectrometry.

As with the previous studies, both algorithms in our model were used to match the glycoform of the wild-type CHO cells in this study. The wild-type glycoform consisted of only three glycans, all of which were complex biantennary types and fucosylated. The distribution of FM3(B1){B1}, FM3(B1){B1}G1, and FM3(B1){B1}G2 glycans was 43%, 42%, and 15% respectively. Both Alg1 and Alg2 produced the same set of calibrated parameter values, shown in Figure 4.11. ManI-II, GnTI-II, and FucT were all at maximum values of 10, with GalT_B1 and GalT_B2 calibrated to smaller values of 6 and 2, respectively. The tendency of GalT ECPs to be significantly lower than the other nonzero ECPs is a trend consistent with all other studies of IgG glycoforms presented in this work (Sealover et al., 2013; Malphettes et al., 2010).

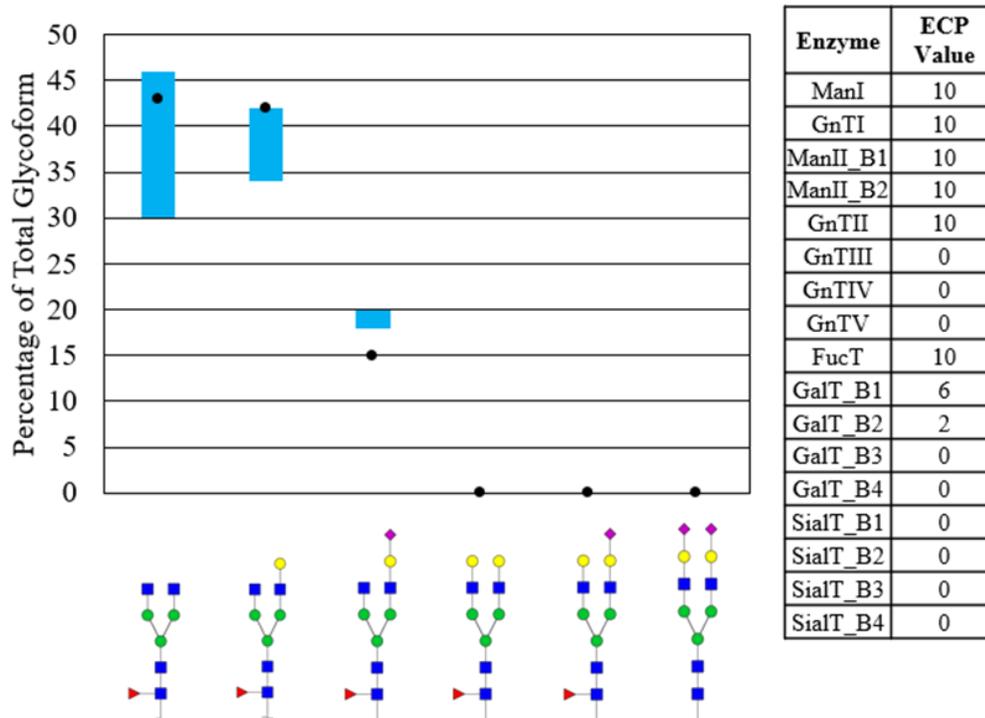
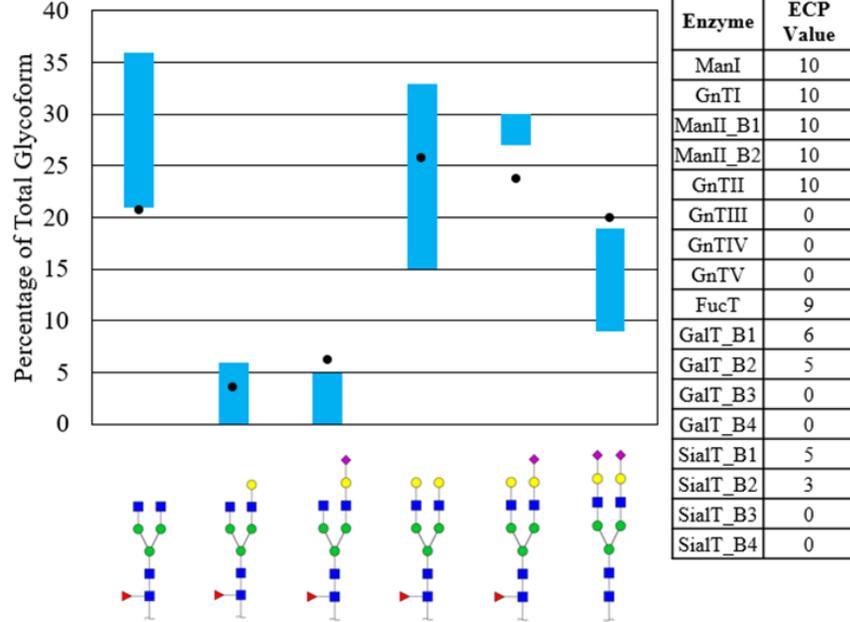


Figure 4.11: Experimental glycoform values (black dots), model prediction ranges (blue bars), and ECP values (Tables) for baseline IgG glycoform.

The stable transfection of *ST6GalII* increased the sialylation percentage of the CHO-K1 glycoform by 69.5%. The glycoform, shown in Figure 4.12, is more heterogeneous than the wild-type glycoform, with six glycans present rather than three. The asialylated glycans consisted of 20.8% FM3(B1){B1}, 3.6% FM3(B1){B1}G1, and 6.2% FM3(B1){B1}G2. The sialylated glycans present were 23.8% M3(B1){B1}G2S2, 25.8% FM3(B1){B1}G2S1, and 19.9% FM3(B1){B1}G2S2. It is important to note that the M3(B1){B1}G2S2 glycan is the only non-fucosylated glycan in both the wild-type and experimental glycoforms. As the experiment was intended to only affect sialylation, lower fucosylation is an

unexpected result with no clear explanation given in the data. Therefore, we used our model to both validate the overexpression experiment as well as provide a potential hypothesis for the ~25% drop in fucosylation.

a.



b.

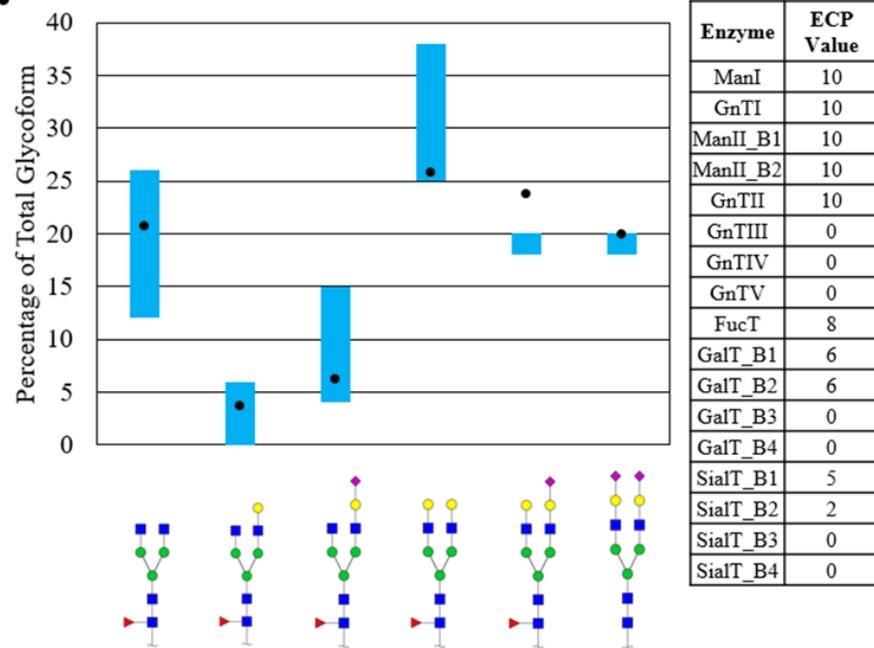


Figure 4.12: Experimental glycoform values (black dots), model prediction ranges (blue bars), and ECP values (Tables) for glycoform after *ST6GalI* overexpression using ECPs calculated with (A) Alg1 and (B) Alg2.

Previous experiments studied with this model were knockouts which eliminated the functionality of a target glycosyltransferase. This was easily simulated in our model by reducing the target ECP to zero. However, for overexpression experiments, it is difficult to determine the magnitude of ECP changes *a priori*. Therefore, rather than increasing the SialT ECP manually, we elected to re-calibrate our model to the experimental glycoform to let our algorithm determine the correct ECP change(s). Not only will this automatically increase the SialT ECP value to the correct level, but it may also change other ECP values to aid in developing a hypothesis for the observed drop in fucosylation.

As with previous simulations, both algorithms were calibrated to the new glycoform to determine which one was more accurate. Alg1 and Alg2 gave unique sets of ECP values, as shown in Figure 4.12. Alg1 (Figure 4.12.a) and Alg2 (Figure 4.12.b) had different values respectively for FucT (9 vs 8), GalT_B2 (5 vs 6), and SialT_B2 (3 vs 2). Examination of the resulting glycoform distributions shows >5% variation in the predicted ranges for all glycans except FM3(B1){B1}G1. As shown, 5/6 glycans are within range for Alg2's ECP values, while Alg1's ECPs result in correct predictions for only 2/6 glycans. Therefore, we used Alg2's ECP values in the ECP analysis.

The ECP values calculated by Alg2 for SialT increased from 0 to 5 (SialT_B1) and 0 to 2 (SialT_B2). This is consistent with the knowledge that the experimental glycoform was measured after overexpressing the corresponding gene *ST6Gall*. Overexpression of *ST6Gall* resulted in an increase in SialT concentration, thus resulting in an increased flux of glycans through all reactions catalyzed by that enzyme in our model and a concurrent increase in the associated ECP values.

Interestingly, we observed a decrease in the ECP value for FucT from 9-8 and an increase in the ECP value of GalT_B2 from 2-6.

The change in FucT flux was expected, as >20% of the experimental glycoform became afucosylated. However, the increase in flux through GalT_B2 was unexpected and was not addressed in the original paper. However, the relationship between the changes in FucT and GalT_B2 are consistent with our knowledge of glycosyltransferase competition. Multiple studies in literature show that the $\beta(1, 4)$ linkage of a galactose residue to a free GlcNAc branch prohibits FucT from binding to the glycan (Kim et al., 2009). If the overexpression of *ST6GalI* triggered an increase in the production of GalT, it is reasonable to assume that the action of FucT may have been inhibited by GalT. This would result in the sudden and unexplained decrease in fucosylated glycans post-transfection. Unfortunately, as the original paper lacked glycosyltransferase expression data, this hypothesis cannot be validated.

Our model was used to match an IgG glycoform with high accuracy (5/6 glycans correctly predicted) as well as validate the effects of *ST6GalI* overexpression on the sialylation percentage of the glycoform. Additionally, we used our model to generate a testable hypothesis for the observed 25% drop in fucosylated glycans.

REFERENCES

- Chui, D., Oh-Eda, M., Liao, Y. F., Panneerselvam, K., Lal, A., Marek, K. W., ... & Marth, J. D. (1997). Alpha-mannosidase-II deficiency results in dyserythropoiesis and unveils an alternate pathway in oligosaccharide biosynthesis. *Cell*, *90*(1), 157-167.
- Dunphy, W. G., Brands, R., & Rothman, J. E. (1985). Attachment of terminal N-acetylglucosamine to asparagine-linked oligosaccharides occurs in central cisternae of the Golgi stack. *Cell*, *40*(2), 463-472.
- Ghaderi, D., Zhang, M., Hurtado-Ziola, N., & Varki, A. (2012). Production platforms for biotherapeutic glycoproteins. Occurrence, impact, and challenges of non-human sialylation. *Biotechnology and Genetic Engineering Reviews*, *28*(1), 147-176.
- Goh, J. S., Liu, Y., Chan, K. F., Wan, C., Teo, G., Zhang, P., ... & Song, Z. (2014). Producing recombinant therapeutic glycoproteins with enhanced sialylation using CHO-gmt4 glycosylation mutant cells. *Bioengineered*, *5*(4), 269-273.
- Herscovics, A. (1999). Importance of glycosidases in mammalian glycoprotein biosynthesis. *Biochimica et Biophysica Acta (BBA)-General Subjects*, *1473*(1), 96-107.
- Houghton, F., & Gleeson, P. A. (2001). Steady-state localization of a medial-Golgi glycosyltransferase involves transit through the trans-Golgi network. *Biochemical Journal*, *358*(1), 33-40.
- Kim, P. J., Lee, D. Y., & Jeong, H. (2009). Centralized modularity of N-linked glycosylation pathways in mammalian cells. *PloS one*, *4*(10), e7317.
- Kim, W. D., Tokunaga, M., Ozaki, H., Ishibashi, T., Honda, K., Kajiura, H., ... & Ohtake, H. (2010). Glycosylation pattern of humanized IgG-like bispecific antibody

produced by recombinant CHO cells. *Applied microbiology and biotechnology*, 85(3), 535-542.

Krambeck, F. J., & Betenbaugh, M. J. (2005). A mathematical model of N-linked glycosylation. *Biotechnology and Bioengineering*, 92(6), 711-728.

Lee, E. U., Roth, J., & Paulson, J. C. (1989). Alteration of terminal glycosylation sequences on N-linked oligosaccharides of Chinese hamster ovary cells by expression of beta-galactoside alpha 2, 6-sialyltransferase. *Journal of Biological Chemistry*, 264(23), 13848-13855.

Malphettes, L., Freyvert, Y., Chang, J., Liu, P. Q., Chan, E., Miller, J. C., ... & Collingwood, T. N. (2010). Highly efficient deletion of FUT8 in CHO cell lines using zinc-finger nucleases yields cells that produce completely nonfucosylated antibodies. *Biotechnology and bioengineering*, 106(5), 774-783.

Onitsuka, M., Kim, W. D., Ozaki, H., Kawaguchi, A., Honda, K., Kajiura, H., ... & Omasa, T. (2012). Enhancement of sialylation on humanized IgG-like bispecific antibody by overexpression of α 2, 6-sialyltransferase derived from Chinese hamster ovary cells. *Applied microbiology and biotechnology*, 94(1), 69-80.

Paquet, M. R., Narasimhan, S., Schachter, H., & Moscarello, M. A. (1984). Branch specificity of purified rat liver Golgi UDP-galactose: N-acetylglucosamine beta-1, 4-galactosyltransferase. Preferential transfer of galactose on the GlcNAc beta 1, 2-Man alpha 1, 3-branch of a complex biantennary Asn-linked oligosaccharide. *Journal of Biological Chemistry*, 259(8), 4716-4721.

Rabouille, C., Hui, N., Hunte, F., Kieckbusch, R., Berger, E. G., Warren, G., & Nilsson, T. (1995). Mapping the distribution of Golgi enzymes involved in the construction of complex oligosaccharides. *Journal of cell science*, 108(4), 1617-1627.

Sealover, N. R., Davis, A. M., Brooks, J. K., George, H. J., Kayser, K. J., & Lin, N. (2013). Engineering Chinese Hamster Ovary (CHO) cells for producing recombinant proteins with simple glycoforms by zinc-finger nuclease (ZFN)—mediated gene knockout of mannosyl (alpha-1, 3-)-glycoprotein beta-1, 2-N-acetylglucosaminyltransferase (Mgat1). *Journal of biotechnology*, *167*(1), 24-32.

Shah, N., Kuntz, D. A., & Rose, D. R. (2008). Golgi α -mannosidase II cleaves two sugars sequentially in the same catalytic site. *Proceedings of the National Academy of Sciences*, *105*(28), 9570-9575.

St Amand, M. M., Radhakrishnan, D., Robinson, A. S., & Ogunnaike, B. A. (2014). Identification of manipulated variables for a glycosylation control strategy. *Biotechnology and bioengineering*, *111*(10), 1957-1970.

Wormald, M. R., Rudd, P. M., Harvey, D. J., Chang, S. C., Scragg, I. G., & Dwek, R. A. (1997). Variations in oligosaccharide– protein interactions in immunoglobulin G determine the site-specific glycosylation profiles and modulate the dynamic motion of the Fc oligosaccharides. *Biochemistry*, *36*(6), 1370-1380.

Xu, X., Nagarajan, H., Lewis, N. E., Pan, S., Cai, Z., Liu, X., ... & Andersen, M. R. (2011). The genomic sequence of the Chinese hamster ovary (CHO)-K1 cell line. *Nature biotechnology*, *29*(8), 735.

Chapter 5

METABOLISM ANALYSIS MODE

5.1 Introduction

While the previous chapter described model operation under the assumption that the level of sugar co-substrate material is constant, this mode assumes that the concentration/activity of all glycosyltransferases are constant. The implication of this assumption is that variations in co-substrate availability become the primary driving force behind changes in glycosylation. This mode can be used to quantify the metabolic flux patterns representative of specific glycoforms based on co-substrate requirements. This is useful for determining preferred network behavior and controlling the glycoform with only media additives (Hills et al., 2001).

This chapter outlines the framework used to model the portion of the CHO cell central-carbon metabolism (CCM) that produces all co-substrates for glycosylation. This chapter begins with an explanation of the network and the generation of an appropriate number of ECPs that can sufficiently describe the network. Next, the algorithms used to calibrate these ECPs to any glycoform will be outlined, followed by a series of four validation tests using media supplementation experiments in literature. For all experiments, raw ECP data is presented in Appendix B.

5.2 Materials and Methods

5.2.1 Reaction Network Development

The reaction network for the CHO cell glycosylation metabolism is smaller than the network in the previous section, as it only covered the conversion of simple sugars (mannose, fucose, glucose, and galactose) to the necessary metabolic precursors for glycosylation (UDP-GlcNAc, UDP-GalNAc, GDP-Fucose, GDP-Mannose, UDP-Galactose, and CMP-NeuAc). The literature was searched extensively to piece together all major portions of the glycosylation metabolism (McAtee et al., 2014; Burleigh et al., 2011; Baker et al., 2000; Butler, 2006). As with the CRN, some assumptions were made here to simplify the network. First, UDP-GlcNAc and UDP-GalNAc are capable of rapid interconversion through only a single reaction. Additionally, UDP-GalNAc does not have any suitable uses in the Golgi for N-glycoform modification. Literature often describes UDP-GlcNAc and UDP-GalNAc as a single UDP-HexNAc pool (Wong et al., 2010) when analyzing glycosylation metabolism. We have therefore combined these metabolites into a single species called UDP-GNAc, and flux from this metabolite represents the transport of UDP-GlcNAc into the Golgi for glycosylation. Secondly, our model does not currently store terminal sugar precursors in the cytosolic pools that are reported in literature. Typically, an excess of co-substrates is produced in the cell, and this excess is often described as co-substrate pools. The relative value of these pool sizes for the experimental cells compared to the control is often reported in literature. However, the lack of quantitative information concerning how many incoming metabolites are

passing into the CHO cell glycosylation metabolism makes an explicit definition of these pool sizes arbitrary, and so our model does not currently include precursor pool storage in its reaction network. As a result, the metabolic portion of our model consists of 38 metabolites and 49 reactions (Figure 5.1).

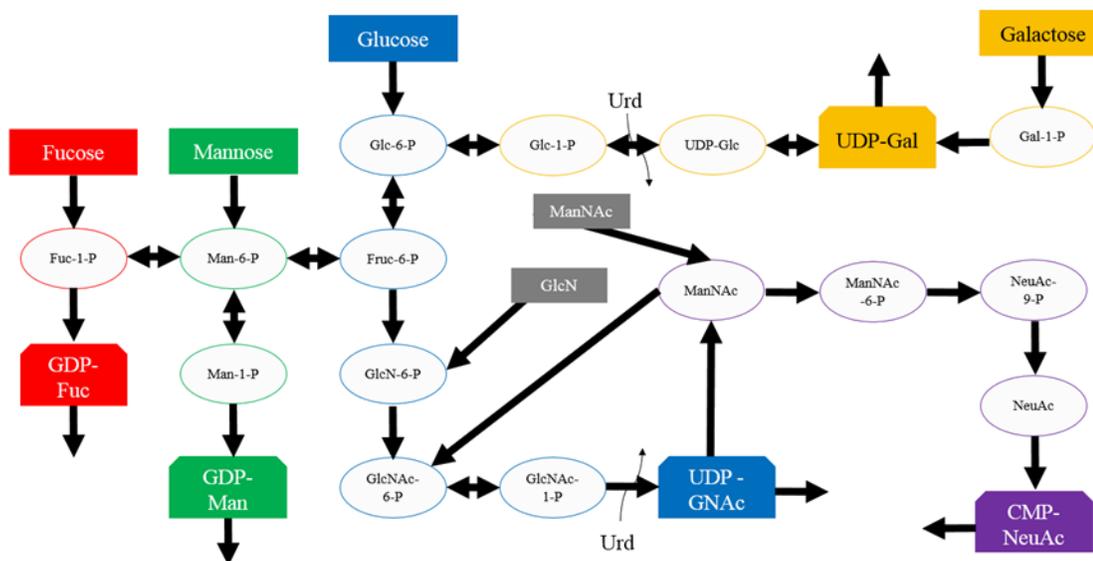


Figure 5.1: Metabolic reaction network for co-substrate production.

5.2.2 Implementation of Parameter-driven Glycosylation Modeling

As with the previous section, a set of parameters were established that describe the reaction network state and are analyzed with constraint-based analysis. These reactions take a steady input flux of six common components/supplements of CHO cell media (glucose, galactose, fucose, mannose, glucosamine (GlcN), and N-acetylmannosamine (ManNAc) and describe their conversion to the six co-substrates for glycosylation (GDP-Man, GDP-Fuc, UDP-Gal, UDP-GlcNAc, and CMP-NeuAc).

This network is composed of only 49 reactions, and is significantly smaller than the glycosyltransferase-mediated reaction network. However, despite its small size, there are more enzymes that catalyze co-substrate formation than glycosyltransferases (McAtee et al., 2014). Additionally, since the glycosyltransferase-mediated reaction network followed an ordered sequence, there is only one way for our model to produce any given glycan. In the metabolism reaction network, the many parallel reactions allow for infinite solutions for any desired set of co-substrates. Due to the larger number of enzymes and an unordered reaction network, care must be taken to establish the correct number of parameters that describe the system completely but do not overspecify the network.

To solve these problems, linear methods were used to identify the minimum number of parameters needed to ensure that any given parameter set has one and only one flux solution. This will ensure our parameters do not underspecify nor overspecify the network. The degrees of freedom of a reaction network is described by Equation 5.1. Here, $[S]$ is an $m*n$ matrix that contains the stoichiometric coefficients for m metabolites and n reactions. The required number of reaction constraints is equal to the degrees of freedom (D.O.F.) of the reaction network, obtained by taking the difference between the rank of $[S]$ and the total number of reactions n_R . From this analysis, we found that 19 constraints are needed to be established for solution distinguishability.

$$D.O.F. = Rank([S]) - n_R \quad (5.1)$$

Although 19 parameters seem like a large number, our reaction network is set to consider reversible reactions as two separate reactions. This means that a parameter which describes a reversible reaction contains information for two constraints. We

have used this information along with our understanding of the CHO cell glycosylation metabolism to establish 13 ECPs that can uniquely describe any metabolic state for sugar co-substrate production. The ECPs as well as the reactions they control can be seen in Figure 5.2. These were chosen as they correspond to key points in the metabolic pathway that are representative of network behavior. The units for all ECPs are in molecules of metabolite/100 glycans. The first six ECPs govern the input flux for all metabolites that are typically supplemented in CHO cell media to alter the glycoform (glucose, galactose, mannose, fucose, ManNAc, and glucosamine). The other seven ECPs dictate the flow of metabolites at key branch points in the pathway. For example, “Glc to Gal” is representative of metabolite flux away from the UDP-GNac synthesis pathway and towards UDP-Gal synthesis.

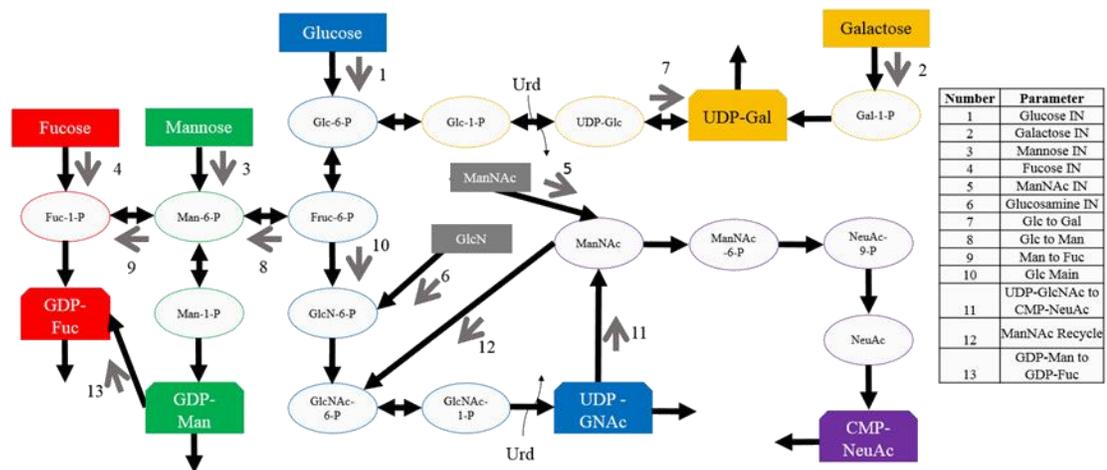


Figure 5.2: Metabolic reaction network for co-substrate production with list of all ECPs (Table) and the corresponding reactions (grey arrows). For ECPs corresponding to bi-directional reactions, the direction of the grey arrow corresponds to the positive direction.

While the ECPs for Enzymatic Analysis Mode are discrete and range from 0-10, the metabolic ECP values are continuous and are exactly equal to the metabolic flux through that reaction. For reversible reactions, we have assigned positive values to reactions moving in the direction shown by the grey arrow in Figure 5.2. For example, the “Man to Fuc” ECP is positive for metabolites flowing from the mannose synthesis pathway (green metabolites in Figure 5.2) to the fucose synthesis pathway (red metabolites in Figure 5.2), and negative if traveling from fucose synthesis to mannose synthesis.

One final consequence of an unordered reaction network is that flux constraints based on custom parameter values entered by the user are extremely difficult to enforce without creating a mass balance error. The ordered glycosyltransferase reactions in the Enzyme Analysis Mode portion of our model allowed for a simple rule for all lower bounds that eliminated any mass balance problems (Chapter 4). However, a simple rule for lower bounds is impossible to create for the metabolic network. As a result, ECPs can only be calculated from an input glycoform (i.e. ‘Mapping Mode’); custom ECPs cannot be entered by the user to simulate a hypothetical distribution of co-substrates (Exploration Mode).

5.2.3 Fitting Algorithm Development

With the reaction network and ECPs defined, we used constraint-based methods and simple algorithms to calculate the correct ECP values for any glycoform entered by the user. As the metabolic ECPs are exactly equal to their corresponding

reactions, no linear transformations were needed to map the parameters to their corresponding reaction flux values. The workflow for calculating all ECPs from any glycoform can be seen in Figure 5.3. First, the quantitative glycoform distribution is transformed into a co-substrate distribution where all mannose, GlcNAc, fucose, galactose, and sialic acid residues in the glycoform are tallied. This represents the set of co-substrate levels required for complete synthesis. Next, constraints are placed on all secretion reactions in the metabolic network for the co-substrates. These constraints are equal to the required sugar co-substrate levels calculated in the previous step, assuming no excess sugars are produced. Finally, FBA was employed to solve the system.

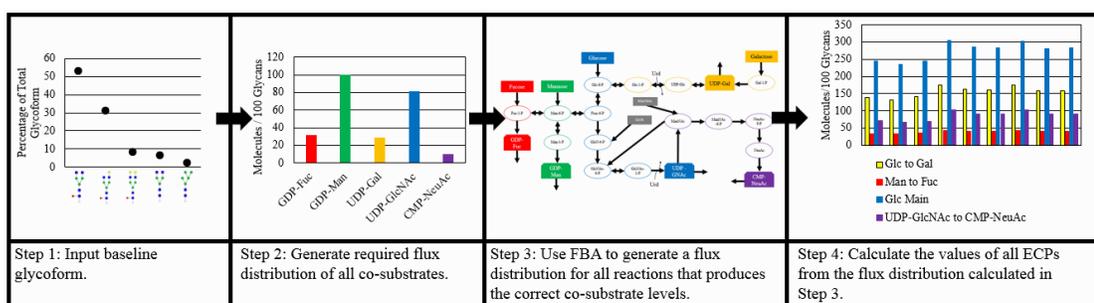


Figure 5.3: Workflow for calculating ECP values from a given glycoform distribution in Metabolism Analysis Mode.

As previously mentioned, there are many solutions for any single flux balance problem for this reaction network. To reduce this space to single solution, the flux balance problem should include a suitable objective function. We have decided to employ a common FBA objective function, referred to as the minimization of internal fluxes (MIF). This objective function is designed based on the assumption that cells

evolve to use their resources as efficiently as possible to grow and divide (Raman & Chandra, 2009). The mathematical description of this assumption is a function that only accepts solutions that minimize the sum of internal flux values in the network. The model solves Equation 5.2, simulating maximal enzymatic efficiency, to obtain an optimal solution for metabolic flux for any glycoform. In this equation, $[S]$ represents the stoichiometric matrix describing the glycosylation metabolism, and v is the vector of steady-state fluxes being solved for. Once this algorithm isolates an optimum solution through the MIF objective function, the appropriate reaction fluxes are used to generate the flux distribution's unique set of ECP values, as explained in the previous section.

$$\min(\sum_{i=1}^n v_i) \text{ subject to } [S]v = 0 \quad (5.2)$$

5.3 Results and Discussion

The methodology of model validation for this section is similar to our validation procedure for Enzyme Analysis Mode in Chapter 4. There are four studies in literature which use different media supplementations to alter the glycoform. To validate our model, we calibrated it to the control and final glycoforms for each experiment to determine if the model's predicted inputs quantitatively matched the corresponding experimental supplementation. This mode of operation can be used to analyze differences in the metabolic flux network due to changes in media composition. This allows the user to gain a quantitative understanding into the CHO cell glycosylation metabolism and, as a result, provides a useful way to control glycosylation through media supplementation.

5.3.1 Wong et al., 2010

This paper aimed to quantify the effect(s) of galactose, glucosamine (GlcN), N-acetylmannosamine (ManNAc), uridine, and cytidine on CHO cell metabolism and glycosylation. Experiments were performed with CHO-Dukx cells adapted to suspension culture and producing human interferon- γ (IFN- γ) proteins. Metabolites were supplemented to cultures in six unique combinations: galactose, galactose + uridine, galactose + ManNAc, ManNAc + cytidine, and GlcN + uridine. Cells were grown in shake-flasks, and all metabolites were fed 48 hrs post-inoculation and grown for another 48 hours before taking metabolite and glycoform measurements.

The control glycoform as well as the glycoform for each experimental condition was used as input data for our model. The ECPs were calibrated to each glycoform and used to validate the qualitative accuracy of our model. It is important to note that for each experimental condition, the user must specify which metabolites (glucose, galactose, ManNAc, GlcN) are being fed to the cells to prevent the algorithm from attempting to import extracellular metabolites that are not present in the experiment. This input is binary (i.e. present vs absent) and does not affect the quantitative results generated by our model. Additionally, the reported glycoform for this paper did not distinguish between fucosylated and afucosylated glycans, nor did they distinguish between glycans with unique extents of galactosylation as long as the number of GlcNAc branches were the same. Therefore, our calculations divided each combined abundance value evenly between the associated glycans.

The ECPs that describe the required input metabolites are shown graphically in Figure 5.4. In this figure, the x-axis represents the different experimental supplementations and the colored bars represent each metabolite required by our

model to produce the corresponding glycoform. The initial results for this simulation qualitatively match all experimental conditions. For each experimental condition, the model correctly predicts the required metabolites using only the glycoform as an input.

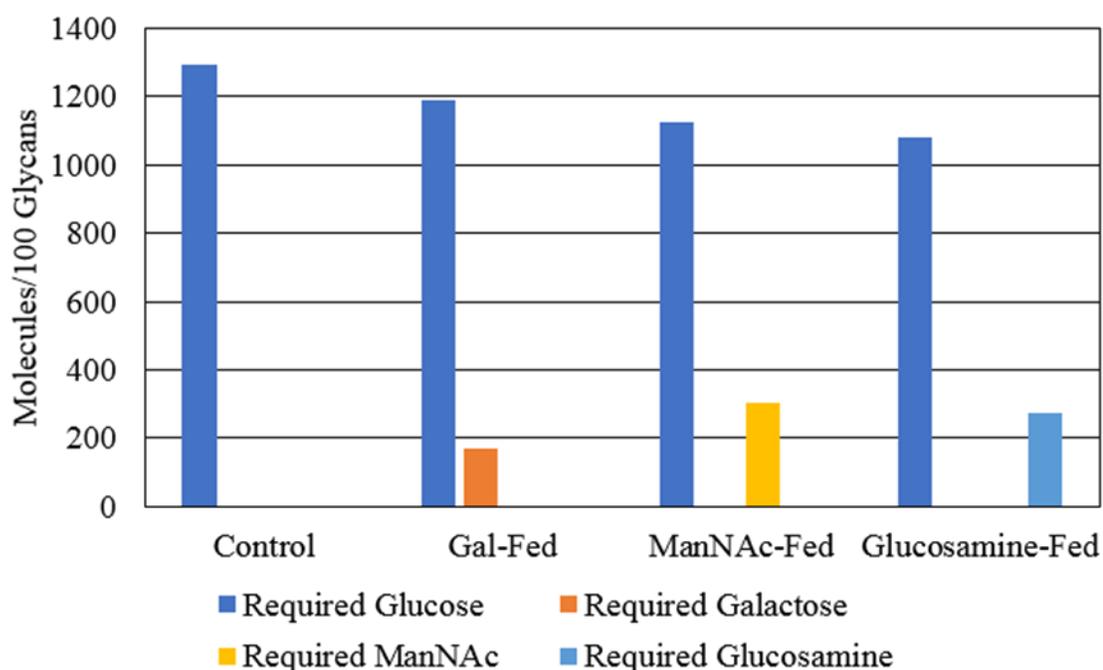


Figure 5.4: Input metabolites (bars) required to match the glycoform measured after culture supplementation of galactose, ManNAc, and glucosamine (x-axis).

We then examined other ECPs to validate that our model can be used for quantitative metabolic analysis consistent with literature findings. As the sialylation extent of glycans is an important aspect of therapeutic efficacy (Ghaderi et al., 2012), we elected to study which conditions most greatly impacted the glycans' extent of sialylation. Figure 5.5 shows the ECP describing flux to CMP-NeuAc (co-substrate for

sialylation) for all experimental conditions. We can see that all cases had a higher extent of sialylation than the control. The conditions that resulted in the highest extent of sialylation were ManNAc (~2 to 2.5-fold increase compared to control) and both ManNAc + cytidine and glucose + uridine (~3 to 3.5-fold increase compared to control). This is likely because ManNAc enters the metabolism along the CMP-NeuAc production pathway (Figure 5.1). Other ManNAc-feeding studies have also reported an increase in sialylation (Gu & Wang, 1998; Baker et al., 2001; Hills et al., 2001). Notably, a synergistic increase in sialylation was found when combining ManNAc and cytidine. As cytidine is a precursor for CMP generation, this effect is consistent with current knowledge of glycosylation metabolism. The addition of glucosamine and glucosamine + uridine also raised the flux of metabolites through the CMP-NeuAc generation pathway by factors of 2.4 and 2.6, respectively. Figure 5.1 shows that glucosamine enters the metabolic pathway prior to the formation of both UDP-GlcNAc and CMP-NeuAc. Supplementation of glucosamine + uridine resulted in a glycoform with increased abundance of biantennary and sialylated glycans by 8% and 30%, respectively, when compared to the control culture. The increase in sialylation may be a result of either an increased CMP-NeuAc availability or an increased concentration of eligible substrates due to the 8% increase of biantennary glycans. However, as the percentage of galactosylated glycans decreased by 14% during this experiment and a free galactose residue is required for sialyltransferase binding, it is likely that the effects are due to an increased availability of CMP-NeuAc through the effect of glucosamine on the CHO cell metabolism.

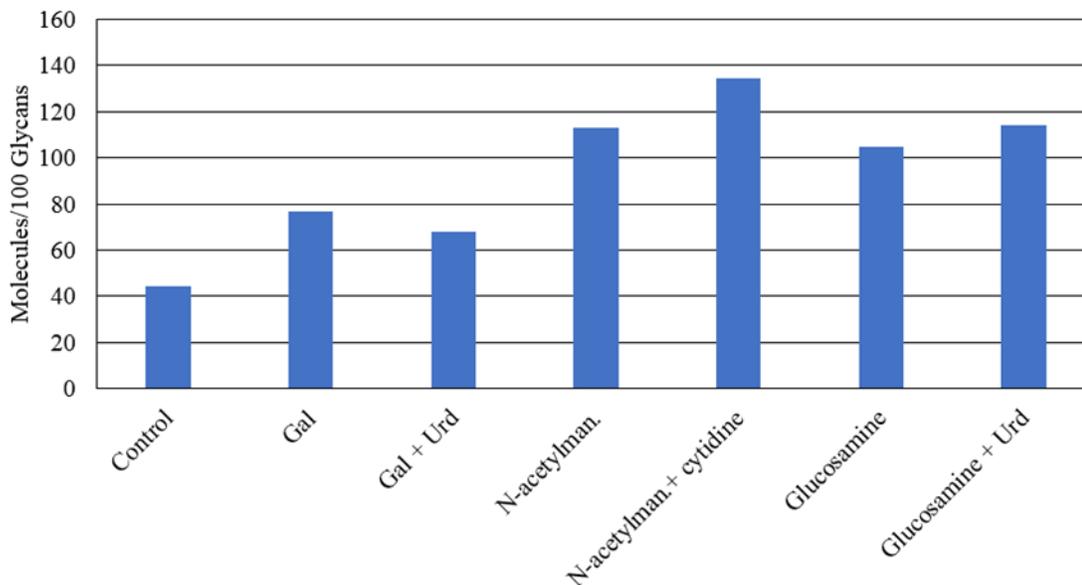


Figure 5.5: Flux of metabolites to CMP-NeuAc generation.

Our model was used to predict the media supplementations necessary to produce experimental glycoforms, and these results were compared with the actual supplementations in literature to validate our model. Our model could correctly predict the addition of glucose, galactose, ManNAc, and glucosamine based on the initial and final glycoform. Additionally, the CMP-NeuAc generation pathway was examined to determine that supplementing the culture with ManNAc and glucosamine resulted in a ~2.5-3.5-fold increase in the extent of sialylation.

5.3.2 Gramer et al., 2011

While the previous study fed various metabolites in unique combinations, this study supplemented the same ratio of galactose, uridine, and manganese to CHO-K1SV cells in increasing levels to maximize galactosylation. Nine different levels of

metabolites were added to different cultures in ratios of 0, 1, 2, 3, 4, 8, 12, 16, and 20 times the bolus concentration of 1 mM uridine, 5 mM galactose, and 0.002 mM MnCl₂. The bolus (referred to as UMG) was added on day three for each culture of CHO cells producing IgG proteins and the glycoform was analyzed via high-performance liquid chromatography (HPLC).

Uridine and Manganese were elected to supplement galactose in these experiments due to their synergistic effects in increasing the extent of galactosylation. While manganese is an essential cofactor for the function of galactosyltransferase (Crowell et al., 2007), uridine does not play a direct role in the production of UDP-Gal (McAtee et al., 2014). However, in this study uridine with galactose was found to provide a synergistic increase in the extent of galactosylation, which was the motivation for this addition. Uridine takes part in a reaction to produce uridine diphosphate (UDP), and thus is essential in creating the correct nucleotide for UDP-Gal and UDP-GNac production (Figure 5.1) (McAtee et al., 2014).

The glycoform for each level of UMG feeding was used as an input for our model to calibrate nine unique sets of ECPs. To quantitatively validate our model, the galactose uptake ECP is shown for each UMG concentration in Figure 5.6. As the amount of fed UMG increases, more galactose molecules are needed to replicate the resulting glycoform. However, the UMG levels exhibit diminishing returns and after approximately 16x UMG feeding, no additional galactose is required to reproduce the corresponding glycoform. These results are in line with conclusions drawn in the original study, where an 18% increase in galactosylation was observed that leveled off post-16x UMG feeding.

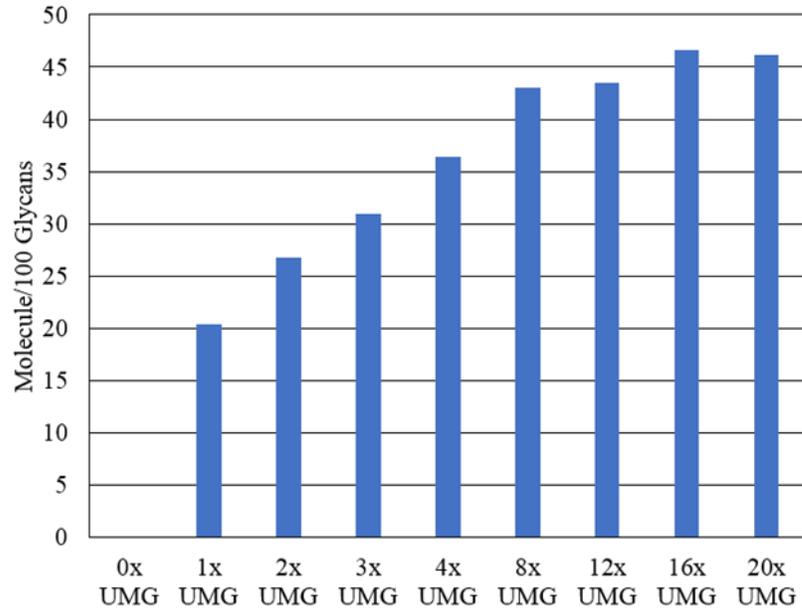


Figure 5.6: Required galactose flux into model as a function of UMG feeding level (units are in fold change of initial bolus concentration: 1 mM uridine, 5 mM galactose, and 0.002 mM MnCl₂).

The study found that UMG feeding was directly responsible for the observed increase in galactose residues in the glycoform. This is consistent with other findings, as direct supplementation of a co-substrate precursor for glycosylation has been shown to increase the presence of that residue on the glycoform (McAtee et al., 2014). Mechanistically, this is most likely due to an increased transport of the final precursor, UDP-Gal, into the Golgi via the reaction network shown in Figure 5.1. There are reactions that convert galactose directly into the co-substrate UDP-Gal, which is then transported to the Golgi and acted on by galactosyltransferase. In our model, the 20x UMG-fed condition exhibits a 2.3-fold increase in the flux through this pathway when compared to the 1x UMG fed flux distribution. It is important to note that glycosylation is limited not only by co-substrate availability, but also by the kinetic

behavior of glycosyltransferases. As the amount of UMG increased, >50% of the glycoform remained free of galactose residues, suggesting that galactosyltransferase was likely saturated.

Increasing the intracellular galactose content had a clear effect on glycosylation. However, over the course of the study, the fucosylation percentage was shown to drop from 100% for 0x UMG-feeding to 94% for 20x UMG-feeding. This implies that the flow of metabolites towards generating GDP-Fucose decreased with increasing UMG-feeding. To determine if the network behavior was altered, we analyzed the other ECPs to generate a hypothesis for the reported drop in fucosylation. Fucosylation requires the enzyme fucosyltransferase to bind to both a suitable substrate as well as the co-substrate GDP-Fucose. As previously mentioned, decreasing fucosylation in the glycoform would result in a lower flow of metabolites into the pathway that generates GDP-Fucose. However, as the metabolic network has multiple branch points, it is of interest to learn if the distribution of metabolites through these branches changed with increased UMG levels. The percentage of material leaving the UDP-GNAc generation branch to produce GDP-Fucose (Figure 5.1), was monitored for each UMG-feeding level, shown in Figure 5.7.

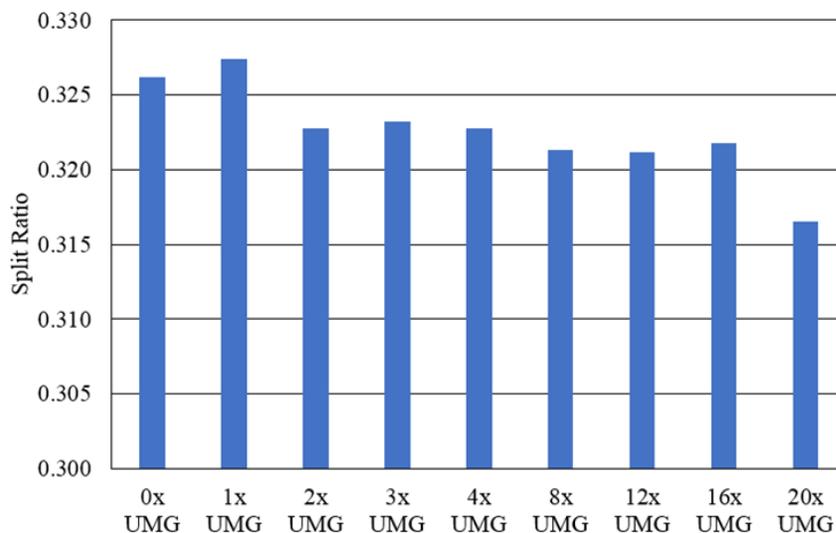


Figure 5.7: Percentage of material used to produce GDP-Fucose as a function of UMG feeding level (units are in fold change of initial bolus concentration: 1 mM uridine, 5 mM galactose, and 0.002 mM MnCl₂).

In Figure 5.7, we observe a small but notable decrease in the split ratio of approximately 2.5% as more UMG is fed. This indicates that the flux of metabolites traveling from glucose to UDP-GNac is increasing relative to GDP-Fucose. It was previously reported that the conversion of GlcNAc-1P to UDP-GlcNAc requires uridine and is located in the main branch of the metabolic pathway (McAtee et al., 2010). As uridine was a metabolite fed to the cells, increasing the amount of uridine may have caused an increased flux from glucose to UDP-GNac and a subsequent reduction in the flux towards GDP-Fucose.

In this study, our model was used to predict the level of galactose needed to match experimental glycoforms from CHO cells supplemented with increasing levels of UMG. We correctly predicted an increasing level of galactose uptake into the glycosylation metabolism and subsequent stagnation after approximately 16x UMG

feeding level. Finally, our model was used to hypothesize that the measured drop in fucosylation may be due to increased activity through reactions consuming uridine, which was indirectly supported by an increased uridine supplementation level in the original paper.

5.3.3 St. Amand et al., 2014

This study analyzed the individual and synergistic effects of galactose, manganese, and ammonia on glycosylation. The combinations studied were manganese, galactose, ammonia, manganese + ammonia, manganese + galactose, galactose + ammonia, and all three together. The goal of this study was to perform a statistical design of experiments (DOE) to elucidate significant effects of various metabolite feeding combinations on the glycoform of CHO cells. They found that manganese increased the abundance of high mannose glycans by 10% and galactose decreased non-galactosylated complex glycans by approximately 15%. We used our model to both validate the conclusions drawn in the article as well as study the metabolic changes associated with each feeding regime.

IgG-producing CHO cells were used for this study with feeding experiments performed in shake flasks using base media that included 5 g/L glucose, 4 mM glutamine, and 25 nM MTX. $MnCl_2$ and Galactose were added at the time of inoculation, and NH_4Cl was added two days after inoculation. The glycoform was measured with MALDI-TOF mass spectrometry and the transcripts of 30 glycosyltransferases were quantified for each feeding regime with qRT-PCR. The co-substrate pools were measured using a previously developed assay (Kochanowski et

al., 2006). In total, glycoforms for eight feeding regimes were performed which encompass all combinations of manganese, galactose, and ammonia in a standard 2^3 factorial design.

To validate our model, the ECPs were calculated for all conditions and the predicted input requirements are shown in Figure 5.8. Our model correctly determines a non-zero amount of galactose uptake in all conditions where galactose was fed in the original study. Interestingly, an approximate 28% reduction in galactose uptake was observed when galactose + manganese was supplemented. Notably, this result contrasts with previous reports of combined manganese and galactose feeding (Gramer et al., 2011). However, we do not know if this reduction is caused by a decrease in UDP-Gal production or by a global decrease in glycoform complexity. If the glycoform as a whole becomes less processed (i.e. lower average antennarity), there would be less substrates available for galactosylation, which would result in the observed reduction in required galactose.

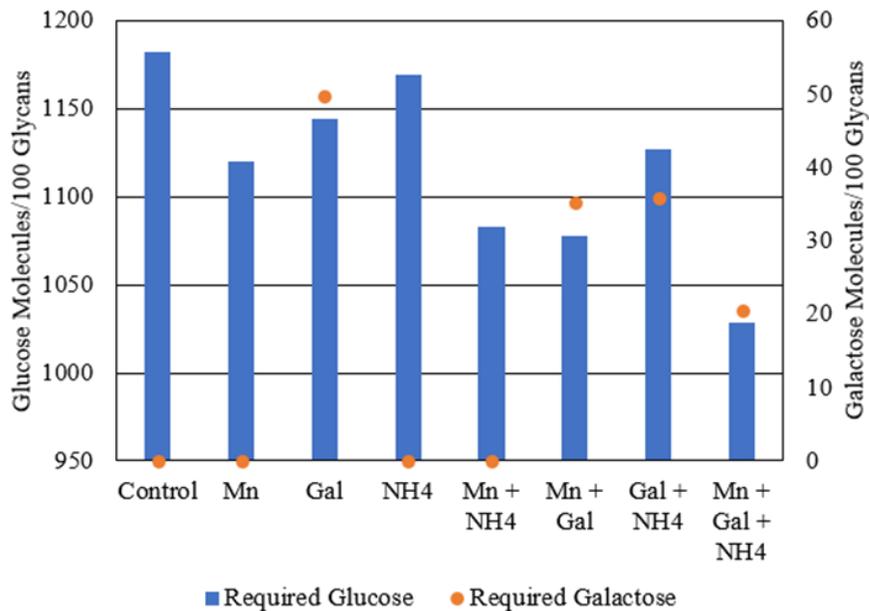


Figure 5.8: Normalized molecules of glucose (left axis, blue) and galactose (right axis, orange) required by our model to produce each reported glycoform.

As glucose is the precursor used to generate the majority of co-substrates, glucose flux into the glycosylation metabolism is a good predictor of glycoform complexity. Figure 5.8 shows that the predicted glucose uptake has changed for each glycoform. Indeed, we observe a decrease in the required glucose by >5% between the galactose-fed culture and the galactose + manganese-fed culture. A further 5% decrease in glucose uptake was observed for the galactose + manganese + ammonia-fed cultures. This supports our hypothesis that the reduction in galactosylation could be due to a global reduction in glycoform complexity rather than a reduction in UDP-Gal abundance. Unfortunately, our analysis is limited due to lack of intracellular glucose data. Extracellular glucose measurements over time were taken but did not appreciably change between these two cultures. However, the paper did report an approximate 20% increase in the M5 (high mannose) glycan between the galactose-fed

cultures and the galactose + manganese-fed cultures, and a further increase of 15% for the galactose + manganese + ammonia-fed cultures. High mannose glycans are the least-processed glycans, and an increase in M5 is indicative of a global reduction in processing.

5.3.4 Burleigh et al., 2011

We have used our model to validate previous experiments in which metabolites were fed that directly affect CHO cell glycosylation metabolism. However, this study aimed to manipulate the metabolic behavior of CHO cells through supplementing the culture with varying amounts of glutamine, a metabolite that is not directly used to create co-substrates for glycosylation. These experiments were performed in shake-flasks and the glucose metabolism, growth rate, ammonia content, and glycoform profile of CHO cells cultivated with various glutamine (Gln) levels (0, 4, and 8 mM) was analyzed. The goal of the paper was to relate metabolic behavior induced by glutamine to changes in the glycoform.

A CHO-K1 cell line expressing human chorionic gonadotrophin (HGC) was used for this study. Cells grown in 0, 4, and 8 mM of glutamine were harvested one, three, and five days after inoculation. Co-substrate concentrations were measured two days after a steady-state was achieved using the assay developed by Kochanowski et al (2006). The glycoform was quantified through successive exposure to a number of glycosidases meant to cleave sialic acid, galactose, fucose, GlcNAc, and mannose residues from the glycan. The cleaved-saccharide concentrations were loaded into an

algorithm known as GlycoBase that calculates the structure of each glycan from the glycosidase data (http://glycobase.nibr.ie:8080/database/show_glycobase.action). However, as this algorithm could not distinguish some glycans, parts of the glycoform were reported as a single value for two glycans. In these cases, we split the value evenly between both glycans when entering the glycoform in our model.

Figure 5.9 shows the required glucose molecules calculated by our model to achieve the glycoform for each condition. We can see that the glucose uptake for 0 mM Gln is approximately 7% lower than the level required for 4 or 8 mM Gln. Additionally, the uptake profiles for the 4 mM and 8 mM Gln cultures are almost identical, suggesting that there is no substantial change of glucose entry into the glycosylation metabolism under these conditions. Over time, the 4 and 8 mM Gln cultures exhibit a comparable decrease in glucose uptake by approximately 3% of the day 1 uptake level for days three and five. Cultures supplemented with no glutamine had a more stable profile over the course of five days, with the glucose uptake decreasing by 1.5% on day three but returning to original levels at day five.

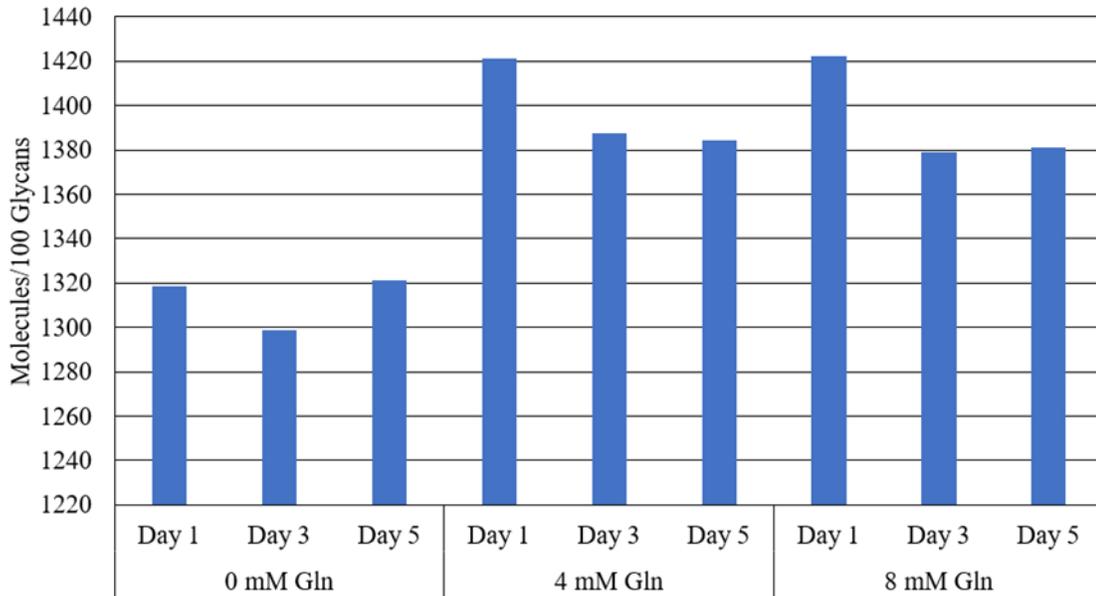


Figure 5.9: Glucose uptake rate required by our model to reproduce the glycoform measured for each experimental condition.

We found that aspects of our glucose consumption predictions for glycosylation quantitatively matched the glucose uptake rates per cell calculated in the study (Figure 5.10). We observe three trends in support of the hypothesis that the glucose uptake rate on a cellular level can control the glucose flux into glycosylation metabolism. First, if we look at the curve for the 0 mM glutamine culture in isolation, we see a decrease in glucose uptake between days 1 and 3, and a subsequent rise on day 5. This qualitatively matches the trend in our model, where the glucose requirement for glycosylation exhibited a sharp decrease followed by an increase over 5 days. The second piece of support for this hypothesis is that the glucose uptake rates for the 4 mM and 8 mM glutamine cultures are almost identical over the course of five days, and the glucose uptake rate was shown to decrease over this time period. This matches the behavior exhibited by our model, as we can see from Figure 5.9 that the

glucose uptake for 4 and 8 mM glutamine cultures are almost identical and decrease over five days.

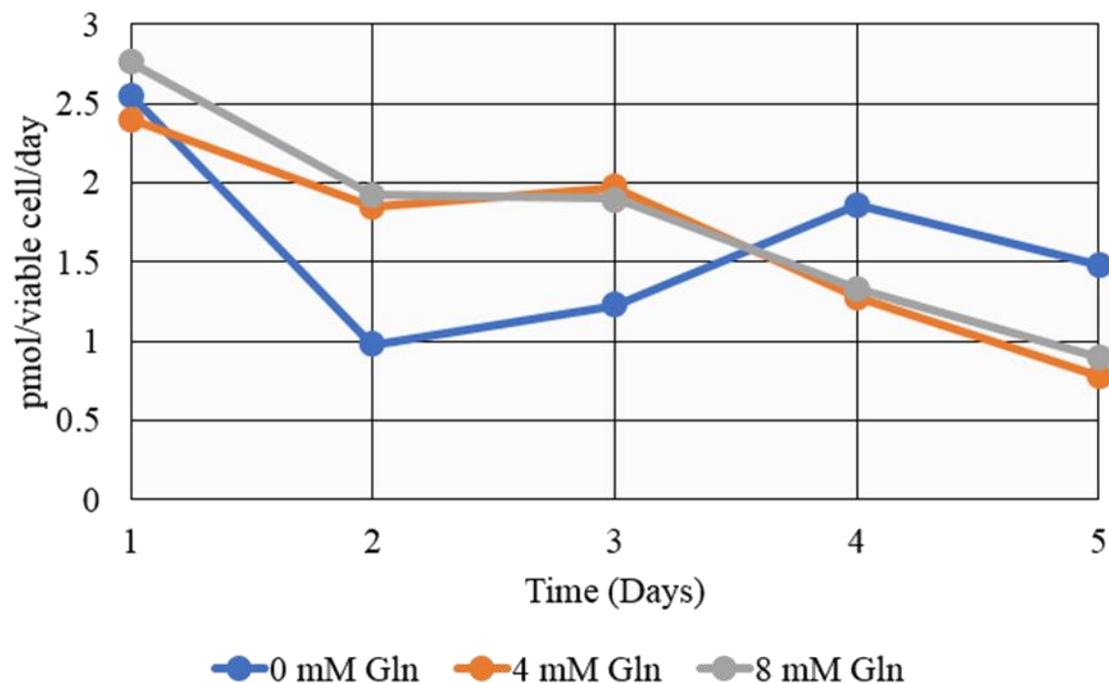


Figure 5.10: Specific glucose uptake rate of experimental CHO cells for 0, 4, and 8 mM Gln over a five-day batch culture (adapted from Burleigh et al.

However, it is clear from comparing our model to the experimental data that glucose uptake cannot be the sole predictor for glycosylation. For example, in Figure 5.10 the glucose uptake rate for all three cultures were almost identical on day 1, yet there was a 7% difference in glucose uptake on day 1 between the 0 and 4/8 mM Gln cultures in our model. Additionally, the final glucose uptake rate for the 0 mM glutamine culture was higher on day five than the uptake rates for both the 4 mM and

the 8 mM cultures. However, our model shows the opposite relationship between the 4 and 8 mM glutamine culture and the 0 mM glutamine culture.

There are two possibilities that can explain the results. The first is that there is a lower amount of glucose entering the glycosylation metabolism in the 0 mM glutamine cultures than for either the 4 or 8 mM glutamine cultures. The principle of glucose availability affecting the glycoform has been explored by Fan et al., who examined the glycoform of CHO cells through varying levels of glucose starvation and observed a 5% increase in the percentage of high mannose glycans between the control culture and the glucose-starved culture (Fan et al., 2015). The second is that a particular subset of reactions within the glycosylation network was disproportionately affected, resulting in a lower required flux of glucose to create the observed glycoform. We can use our model to determine which hypothesis is more likely by comparing the other ECPs that describe the rest of the reaction network.

Our reaction network can be divided into four sub-branches that describe the flux of metabolites to each co-substrate (UDP-GlcNAc, CMP-NeuAc, GDP-Fuc, and UDP-Gal) that is used to modify the glycoform. These sub-branches can be seen in Figure 5.11, where each color represents a different sub-branch that corresponds to the final co-substrate product; the ECP associated with each sub-branch is labeled accordingly. These ECPs can be analyzed for each experimental condition over time in a similar manner to our analysis for glucose uptake. If the behavior of any ECP does not match the qualitative behavior of glucose uptake for the corresponding experimental dataset, it suggests that there is an additional stress on the glycosylation pathway beyond a global reduction in glucose flux. However, if all ECPs behave in a

similar way to the global glucose uptake ECP, it is likely that glucose uptake is the primary source of the observed glycoform changes.

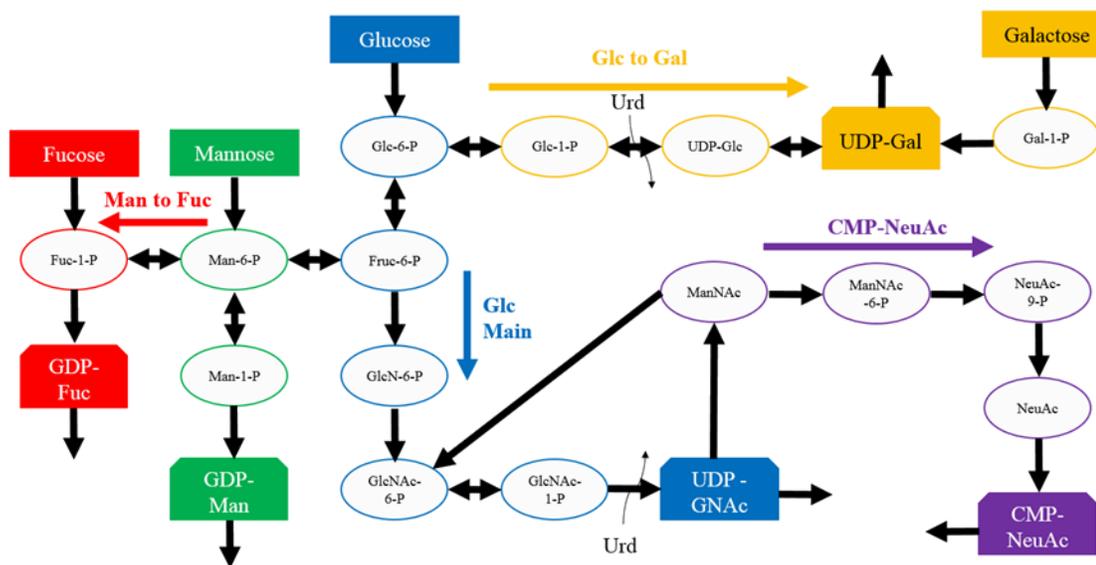


Figure 5.11: Metabolic reaction network for co-substrate production marked with reactions used to calculate the ECPs 'Man to Fuc' (red), 'Glc to Gal' (yellow), 'Glc Main' (blue) and 'CMP-NeuAc' (purple).

Figure 5.12 shows the ECP values for each glycosylation sub-branch in our model. Analysis of each sub-branch shows almost complete qualitative consistency when compared to the glucose uptake values in Figure 5.9. Each sub-branch exhibits a small drop of 2-7% followed by an increase of 3-8% for the 0 mM glutamine culture. For both the 4 and 8 mM cultures, a decrease of 5-9% in all sub-branch fluxes is observed between days 1 and 3, which remains stable (flux changes from 0-2%) on day 5. All sub-branches exhibit percent changes within 5% of each other, and no sub-branch exhibits a change in activity that qualitatively contrasts with the behavior of

Gln uptake in Figure 5.9. Consequently, our model suggests that the changes observed in glycosylation are not due to an internal change in a sub-branch, rather the cause is likely due to changing glucose flux into glycosylation metabolism which affects all sub-branches equally. Other factors that have precedent in contributing to the glycoform, such as ammonia concentration (Yang & Butler, 2000), were quantified in this paper. However, no discernable correlations were found in the data that could generate a hypothesis for the decrease in glucose flux that is consistent for all data points.

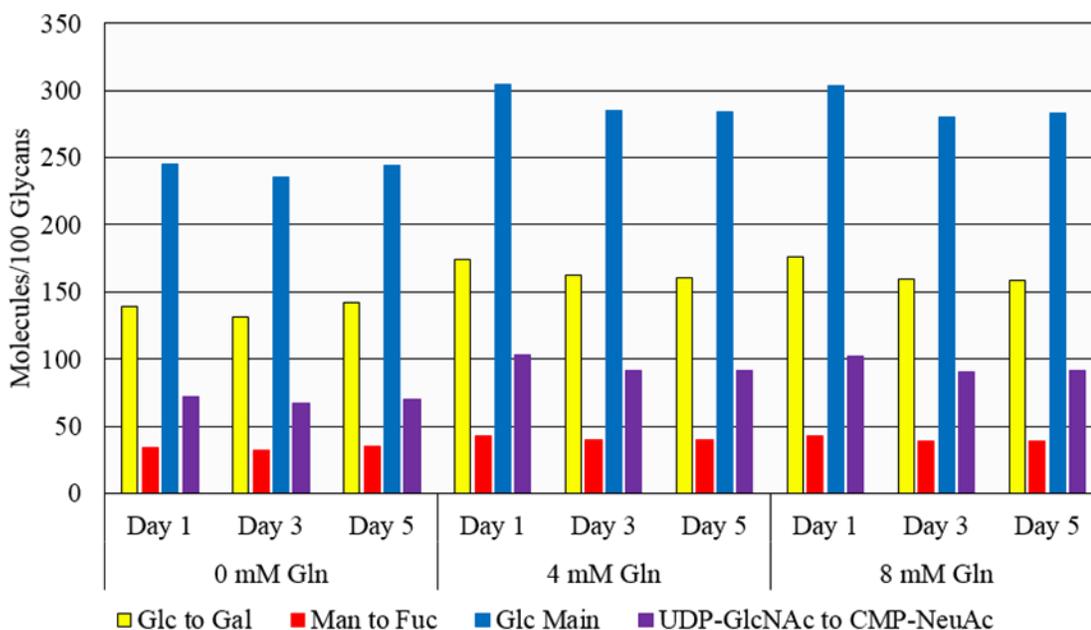


Figure 5.12: Distribution of ECP values calculated from glycoform measured on days 1, 3, and 5 of culture supplemented with 0, 4, and 8 mM Gln.

Our model was used to study the effect(s) cellular glucose consumption has on the glycoform by quantifying the required glucose flux into glycosylation metabolism and comparing it with cellular glucose uptake rates in cultures supplemented with different levels of glutamine. Trends in glucose consumption qualitatively matched the trends predicted by our model for cultures over time, however the data was quantitatively inconsistent and did not conclusively determine the cause for the observed changes in glycosylation. Our model was then used to determine if the glycosylation changes were due to a global reduction in glucose flux into glycosylation metabolism or if a specific set of reactions were impacted through Gln addition. We were able to provide support for the hypothesis that the observed changes were due to a global reduction in glucose flux rather than an inhibition of a subset of reactions in glycosylation through analysis of the ECP values for each experimental condition. Such information is invaluable for future experiments towards understanding the relationships between metabolism and glycosylation.

REFERENCES

- Baker, K. N., Rendall, M. H., Hills, A. E., Hoare, M., Freedman, R. B., & James, D. C. (2001). Metabolic control of recombinant protein N-glycan processing in NS0 and CHO cells. *Biotechnology and bioengineering*, *73*(3), 188-202.
- Burleigh, S. C., van de Laar, T., Stroop, C. J., van Grunsven, W. M., O'Donoghue, N., Rudd, P. M., & Davey, G. P. (2011). Synergizing metabolic flux analysis and nucleotide sugar metabolism to understand the control of glycosylation of recombinant protein in CHO cells. *BMC biotechnology*, *11*(1), 95.
- Butler, M. (2006). Optimisation of the cellular metabolism of glycosylation for recombinant proteins produced by mammalian cell systems. *Cytotechnology*, *50*(1-3), 57.
- Crowell, C. K., Grampp, G. E., Rogers, G. N., Miller, J., & Scheinman, R. I. (2007). Amino acid and manganese supplementation modulates the glycosylation state of erythropoietin in a CHO culture system. *Biotechnology and bioengineering*, *96*(3), 538-549.
- Fan, Y., Jimenez Del Val, I., Müller, C., Lund, A. M., Sen, J. W., Rasmussen, S. K., ... & Andersen, M. R. (2015). A multi-pronged investigation into the effect of glucose starvation and culture duration on fed-batch CHO cell culture. *Biotechnology and bioengineering*, *112*(10), 2172-2184.
- Gramer, M. J., Eckblad, J. J., Donahue, R., Brown, J., Shultz, C., Vickerman, K., ... & van Berkel, P. H. (2011). Modulation of antibody galactosylation through feeding of uridine, manganese chloride, and galactose. *Biotechnology and bioengineering*, *108*(7), 1591-1602.

Gu, X., & Wang, D. I. (1998). Improvement of interferon-g sialylation in Chinese hamster ovary cell culture by feeding of N-acetylmannosamine. *Biotechnology and bioengineering*, 58(6), 642-648.

Hills, A. E., Patel, A., Boyd, P., & James, D. C. (2001). Metabolic control of recombinant monoclonal antibody N-glycosylation in GS-NS0 cells. *Biotechnology and Bioengineering*, 75(2), 239-251.

Kochanowski, N., Blanchard, F., Cacan, R., Chirat, F., Guedon, E., Marc, A., & Goergen, J. L. (2006). Intracellular nucleotide and nucleotide sugar contents of cultured CHO cells determined by a fast, sensitive, and high-resolution ion-pair RP-HPLC. *Analytical biochemistry*, 348(2), 243-251.

McAtee, A. G., Templeton, N., & Young, J. D. (2014). Role of Chinese hamster ovary central carbon metabolism in controlling the quality of secreted biotherapeutic proteins. *Pharmaceutical Bioprocessing*, 2(1), 63-74.

Raman, K., & Chandra, N. (2009). Flux balance analysis of biological systems: applications and challenges. *Briefings in bioinformatics*, 10(4), 435-449.

St Amand, M. M., Radhakrishnan, D., Robinson, A. S., & Ogunnaike, B. A. (2014). Identification of manipulated variables for a glycosylation control strategy. *Biotechnology and bioengineering*, 111(10), 1957-1970.

Wong, N. S., Wati, L., Nissom, P. M., Feng, H. T., Lee, M. M., & Yap, M. G. (2010). An investigation of intracellular glycosylation activities in CHO cells: effects of nucleotide sugar precursor feeding. *Biotechnology and bioengineering*, 107(2), 321-336.

Yang, M., & Butler, M. (2000). Effects of ammonia on CHO cell growth, erythropoietin production, and glycosylation. *Biotechnology and bioengineering*, 68(4), 370-380.

Chapter 6

DEVELOPMENT OF MATLAB GUI

6.1 Introduction

Our model consists of multiple Matlab scripts and functions that combine glycan and reaction data stored in Microsoft Excel sheets with the COBRA constraint-based modeling engine in Matlab. Each component was designed to be as modular as possible to maximize usability and alterability. However, without an intuitive GUI, our model still has an extremely high learning curve for new users. Successful operation of the model would require simultaneous editing of both Matlab scripts and the associated Excel sheets, making de-bugging and error-free operation difficult.

To lower the learning curve on our tool, a comprehensive GUI was designed in Matlab that executes the model's main functions. Although all model functions can be handled from Matlab scripts directly, the GUI allows the user to change input data, run simulations, and extract results for model functions in one place. This section provides a comprehensive overview of all Matlab functions and GUI capabilities included in our model. All source code will be displayed in Appendix C.

6.2 Features

The GUI (Figure 6.1) is a single panel interface which can be divided into four major sections. Section 1 is located in the upper left and dictates what mode of operation under which the user would like to run simulations. Section 2 contains algorithms used to run the model, including commands to obtain a glycoform from a

set of ECPs (Exploration Mode) or obtain a set of ECPs from a glycoform (Mapping Mode). Section 3 opens up a dialogue box for the user to look up chemical and physical attributes of any glycan in the model. Finally, Section 4 contains an area where the user will enter the required parameters or glycoform as well as view the results of the simulation.

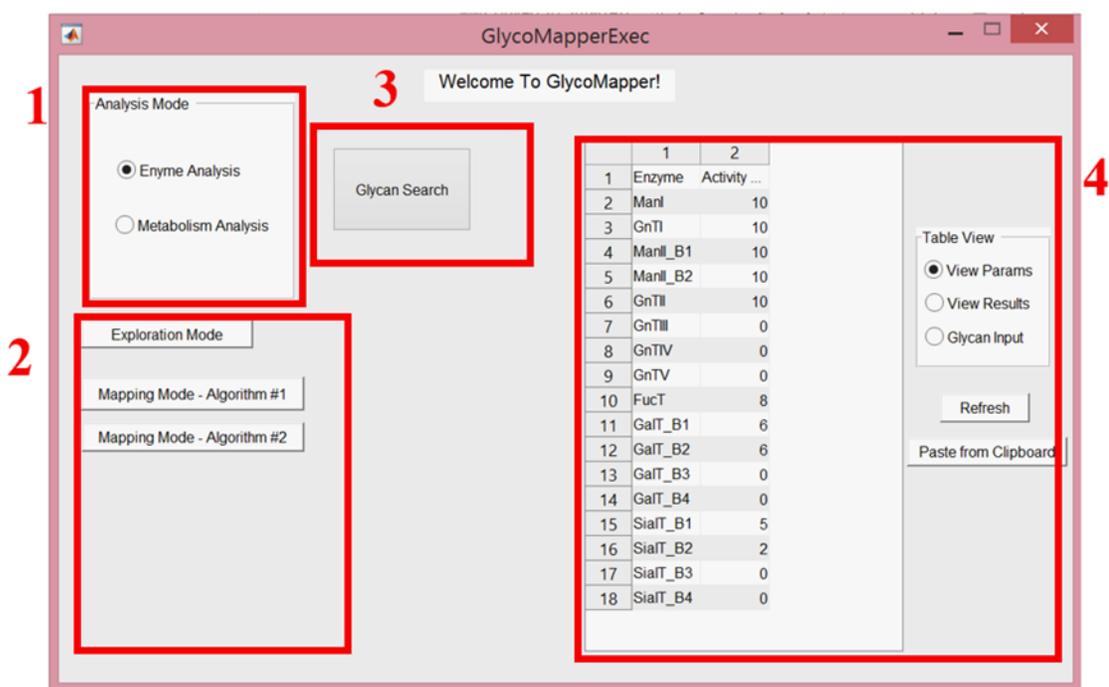


Figure 6.1: GUI panel with labelled sections: 1) Mode of analysis, 2) Algorithm selection and setup, 3) Glycan search tool, and 4) Data input and results table.

For Section 1, there are two possible modes that the user can select: Enzyme Analysis Mode and Metabolism Analysis Mode. There are separate scripts written that perform constraint-based analysis for the glycosyltransferase reaction network or the

metabolism reaction network, depending on which mode is active. If Enzyme Analysis Mode is selected, the visible buttons in Section 2 are the same as those shown in Figure 6.1. Enzyme Analysis Mode operates model functions according to the methods described in Chapter 4 to analyze changes in glycosyltransferase concentration/activity, and Metabolism Analysis Mode is explained and used in Chapter 5 to study co-substrate generation.

Each button in Section 2 calls on a specific algorithm and shows the results in Section 3's display table. There are two ways to use our model to study glycosylation: simulating a glycoform from a set of ECPs entered by the user or calculating the optimal set of ECPs from a user-submitted glycoform. Each button visible in Section 2 under Enzyme Analysis Mode executes one of those two functions. Exploration Mode calculates a hypothetical glycoform given a set of ECP values while Mapping Mode finds an optimal set of parameters given a glycoform. There are two algorithms available for Mapping Mode, denoted as Algorithm 1 and Algorithm 2, respectively. These algorithms calibrate the relevant ECPs to a user-submitted glycoform, and the assumptions and procedures for each algorithm differ (Chapter 4. 2. 4). If Metabolism Analysis Mode is selected, the GUI layout changes into that shown in Figure 6.2. This introduces a new panel in the lower left corner of the GUI which is necessary for metabolic analysis. This Feeding Panel tells the model which metabolites to consider as a significant media component. The user should check the boxes that correspond to the media supplements used in the experiment. Lastly, as this mode only has one available fitting algorithm, the Exploration Mode as well as the second Mapping Mode buttons are removed from view.

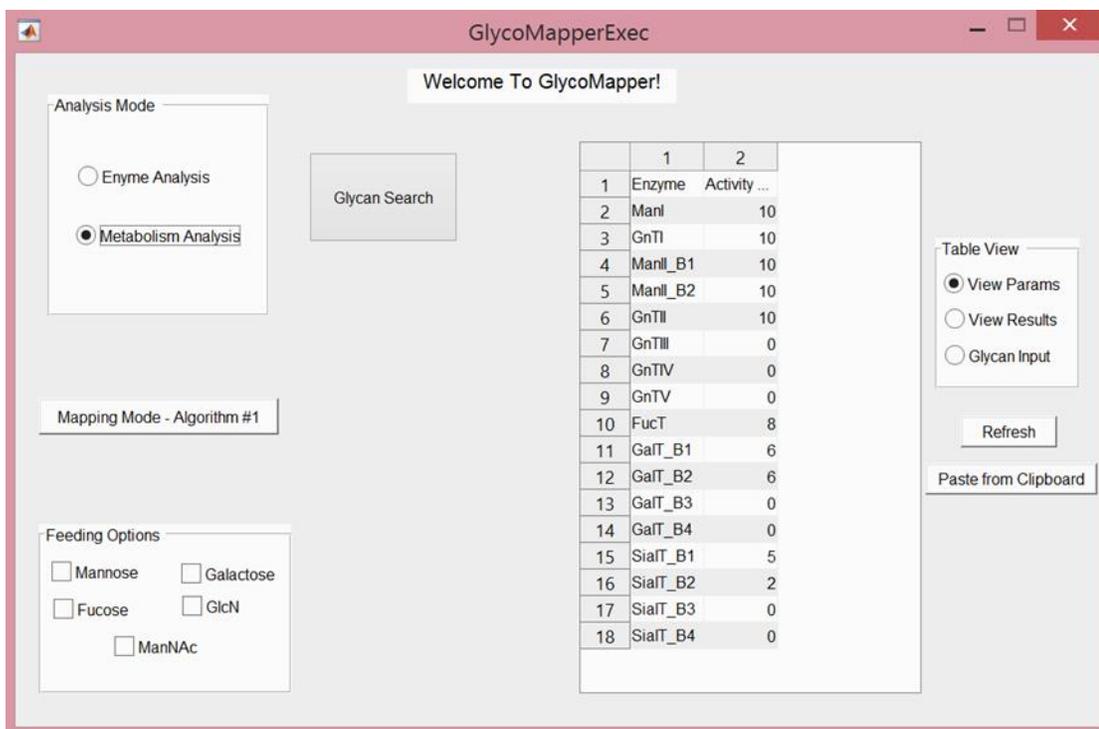


Figure 6.2: GUI when Metabolism Analysis Mode is selected.

Section 3 is designed to give the user detailed information for all glycans included in the model. The Glycan Search function allows the user to select a glycan and view information on the chemical formula, KEGG ID (Kanehisa et al., 2017; Kanehisa et al., 2016; Kanehisa & Goto, 2000), PubChem ID (Kim et al., 2015), International Chemical Identifier (InChI) string (Heller et al., 2013), and SMILES string (Anderson et al., 1987). This button opens a separate glycan selection pane where the user can find this information on any glycan. The information is stored on the corresponding Excel sheet, and edits/additions can be made by the user at any time by modifying the appropriate Excel fields. This functionality is for reference purposes only, and does not affect any algorithms.

The last section contains the GUI's display window. This area is where the user inputs all parameter values and/or retrieves the results. The box to the right of the display window toggles which data the user views. The View Params option is where the user can enter or view ECP values. View Results shows the user all secreted glycan flux values for a given simulation. Glycan Input is where the user would input a glycoform prior to using Mapping Mode. The Refresh button is used whenever the user toggles the Table View options to show the new screen, and the Paste from Clipboard button is used to paste data in the clipboard to the GUI table.

The model's GUI offers an easy-to-use interface for model utilization and configuration. This is an advantage over the original Glyco-Mapper developed in the Lee lab (Kremkow & Lee, submitted), which places the user interface, model calculations, glycan data, and reaction network all on the same Excel file. This makes the learning curve for the Glyco-Mapper extremely high for inexperienced users. In contrast, our model can be quickly used to study the glycoform with relatively little instruction. Additionally, for experienced users, the modular system makes it easier to make changes to the source code, reaction networks, or GUI functions.

REFERENCES

Anderson, E., Veith, G. D., & Weininger, D. (1987). *SMILES, a line notation and computerized interpreter for chemical structures*. US Environmental Protection Agency, Environmental Research Laboratory.

Heller, S., McNaught, A., Stein, S., Tchekhovskoi, D., & Pletnev, I. (2013). InChI-the worldwide chemical structure identifier standard. *Journal of cheminformatics*, 5(1), 7.

Kanehisa, M., Furumichi, M., Tanabe, M., Sato, Y., & Morishima, K. (2017). KEGG: new perspectives on genomes, pathways, diseases and drugs. *Nucleic acids research*, 45(D1), D353-D361.

Kanehisa, M., & Goto, S. (2000). KEGG: kyoto encyclopedia of genes and genomes. *Nucleic acids research*, 28(1), 27-30.

Kanehisa, M., Sato, Y., Kawashima, M., Furumichi, M., & Tanabe, M. (2016). KEGG as a reference resource for gene and protein annotation. *Nucleic acids research*, 44(D1), D457-D462.

Kim, S., Thiessen, P. A., Bolton, E. E., Chen, J., Fu, G., Gindulyte, A., ... & Wang, J. (2015). PubChem substance and compound databases. *Nucleic acids research*, 44(D1), D1202-D1213.

Kremkow, B., Lee, K. (2017) (Submitted)

Chapter 7

CONCLUSIONS AND FUTURE WORK

7.1 Conclusions

This paper established and developed a constraint-based framework for the quantitative analysis of glycosylation using a significantly lower number of parameters than previous glycosylation models. Our description of glycosylation as a network of fluxes allowed the model to be easily calibrated to the majority of N-glycoforms produced in CHO cells. Additionally, we have shown that the parameter system implemented in both the Enzyme Analysis and Metabolism Analysis Modes of our model can be used to predict changes in the glycoform after overexpression and knockout experiments as well as study variations in the glycosylation metabolism due to media changes.

When run under Enzyme Analysis Mode, our model was able to successfully calibrate to four baseline glycoforms with 4/5, 10/17, 6/7, and 2/3 glycans quantitatively matched correctly. However, the majority of incorrectly-calibrated glycans were missed by less than 5%. Our model was able to correctly predict the quantitative effects of an *MgatI* and *FUT8* knockout through manipulation of the appropriate ECPs. Lastly, we have shown how our model can be used to generate testable hypothesis for unexpected changes in the glycoform through ECP calibration to both the initial and final glycoform.

Metabolism Analysis Mode allowed our model to simulate the metabolic changes associated with media supplementation by calibrating our model to the initial and final glycoform. Our model was shown to require the correct media

supplementations for all experiments. Additionally, our model showed that certain metabolic additives such as uridine had a significant effect on glycoform antennarity, a phenomenon that was validated by our knowledge of glycosylation metabolism. Though this mode could not make predictions, it was able to successfully create hypothesis for observed glycoform changes as a result of media supplementation.

Our model's accuracy in replicating experiments from literature prompted us to design a custom-fitting algorithm, a comprehensive glycan database, and a compact GUI to maximize usability and alterability. The fitting algorithms allow the user to generate accurate ECP estimates for any incoming glycoform. The glycan database includes all available KEGG and PubChem I.D. numbers, chemical formulas, and SMILES keys for select glycans in our model. Lastly, the GUI allows the user to run all necessary model operations from one place quickly and easily. This thesis has laid the groundwork for a model that can successfully predict and study CHO cell glycosylation without requiring any traditional kinetic parameters. The model's accuracy, simplicity, and flexibility provide a useful tool with which to study glycosylation to improve the efficiency of biopharmaceutical manufacturing.

7.2 Future Work

Given more time, this thesis can be expanded in two directions. The first is to improve upon the constraint-based modeling methods presented in this paper to increase accuracy, and the second is to develop a method that allows our model to predict co-substrate distributions while in Metabolism Analysis Mode.

7.2.1 Improvement of Constraint-based Modeling for Glycosylation

Chapter 4 showed how we can assign ECPs to all enzymes for N-glycosylation to match experimentally-derived glycoforms. However, in instances where the glycoform was highly heterogeneous, the model exhibited decreased accuracy (9/17 glycans correctly matched) when compared to simple IgG glycoforms (4/5 glycans correctly matched). Additionally, model output for all experiments included a large number of glycans in the 0-10% range (Appendix A). Due to the nature of low-parameter constraint-based analysis, our model will naturally have lower precision than a kinetic model. However, improved ECP discretization and approximate enzyme kinetics may be able to improve model precision (range of predicted values for each glycan) and accuracy (agreement between model prediction and experiment) while keeping parametric and computational demands low (Sokhansanj et al., 2009).

The value of a fuzzy (continuous) parameter in literature is described as a Gaussian window function (Du et al., 2005). The window function can be modified to describe a broad or narrow distribution based on parameter certainty. Though each discrete ECP in our model is mapped to a continuous flux distribution with upper and lower bounds, these bounds behave in a similar manner to the window function in that they can be modified to reflect parameter certainty. These bounds can be increased by lowering the number of ECPs that are mapped to the 0-100% flux range. This would increase model accuracy (defined as the number of glycans correctly predicted), but lower precision (model output would have larger predicted ranges for each glycan). Inversely, increasing the number of ECPs that are mapped to the flux range would result in decreased magnitude of upper and lower bounds. This would lower accuracy

but increase precision. This can be tuned to optimize the accuracy and precision for model calibration.

As a consequence of glycosylation's large-scale reaction network, the majority of glycosyltransferases are capable of binding to many different glycans. In kinetic models, parameters such as the maximal rate velocity and equilibrium constant play a role in determining the final composition of the glycoform. However, no such structure exists in our model, and a large number of glycans (referred to as noise glycans) are present in small amounts (0-10%) for any given ECP distribution. Structural kinetic modeling is an extension of constraint-based analysis that aims to introduce kinetic effects in a reaction network while minimizing the number of required parameters (Steuer et al., 2006). These kinetic effects are represented as a linear transformation of the stoichiometric matrix to account for preferential reaction pathways. Such methodology can be implemented in our model based on our current understanding of *in vivo* enzymatic reaction velocities. This can further constrain the network to eliminate the presence of unwanted intermediate glycans in the final result.

7.2.2 Implementing Predictive Functions in Metabolism Analysis Mode

In Chapter 5, we demonstrated how Metabolism Analysis Mode can validate the requirement for media additives such as galactose, uridine, glucose, glucosamine, and N-acetylmannosamine using the glycoform as the sole input. However, there is currently no way for our model to predict co-substrate production changes from a set of media changes *a priori*. This step is critical in our overall goal of developing a

unified glycosylation model that combines the co-substrate generation network and the glycosyltransferase reaction network.

The biggest challenge associated with predicting co-substrate production in our metabolic reaction network is keeping our ECP constraints from breaking the mass balance. To generate a final co-substrate distribution from a set of ECPs, a more sophisticated method of constraining the reaction network should be implemented. One way literature has introduced more sophisticated control over a constraint-based model is known as reaction coupling (Basler et al., 2016). The principle behind reaction coupling is that in metabolic networks, many reactions are quantitatively related. For example, a single equilibrium reaction consists of both a forward and reverse reaction. Mathematically, these reactions are said to be directionally coupled (when one is non-zero, the other must be zero).

The principle of reaction coupling can be applied to our network to ensure that our imposed constraints do not break the mass balance. Rather than an ECP forcing an absolute value of material through the reaction, select ECPs can be coupled together to describe flux as a percentage of flow through the corresponding node. In this way, any combination of ECPs can be specified by the user and the model will successfully predict the resulting co-substrate distribution.

REFERENCES

Basler, G., Nikoloski, Z., Larhlimi, A., Barabási, A. L., & Liu, Y. Y. (2016). Control of fluxes in metabolic networks. *Genome research*, 26(7), 956-968.

Du, P., Gong, J., Wurtele, E. S., & Dickerson, J. A. (2005). Modeling gene expression networks using fuzzy logic. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 35(6), 1351-1359.

Sokhansanj, B. A., Datta, S., & Hu, X. (2009). Scalable dynamic fuzzy biomolecular network models for large scale biology. In *Fuzzy Systems in Bioinformatics and Computational Biology* (pp. 235-255). Springer, Berlin, Heidelberg.

Steuer, R., Gross, T., Selbig, J., & Blasius, B. (2006). Structural kinetic modeling of metabolic networks. *Proceedings of the National Academy of Sciences*, 103(32), 11868-11873.

Appendix A

RAW MODEL OUTPUT (CHAPTER 4 RESULTS)

A.1 Preface

This chapter contains the raw model output for all experiments analyzed in Chapter 4. In all tables, reactions without a reactant indicate entry reactions into the network, and reactions without a product indicate secretion reactions out of the network.

A.2 Chapter 4.3.1 Raw Output

Table A.1: (Chapter 4.3.1) Model output after calibration to control glycoform.

Reaction	Min	Max
-> M9[g]	100	100
M9[g] -> M8[g]	90	100
M8[g] -> M7[g]	90	100
M7[g] -> M6[g]	90	100
M6[g] -> M5[g]	90	100
M5[g] -> M5(B1)[g]	90	100
M5(B1)[g] -> M4(B1)[g]	90	100
M4(B1)[g] -> M3(B1)[g]	90	100
M3(B1)[g] -> M3(B1){B1}[g]	90	100
M3(B1){B1}[g] -> FM3(B1){B1}[g]	90	100
M3(B1)[g] -> FM3(B1)[g]	0	10
M5(B1)[g] -> FM5(B1)[g]	0	10
M4(B1)[g] -> FM4(B1)[g]	0	10
M3(B1){B1}[g] -> M3(B1){B1}G1[g]	0	10
FM3(B1){B1}[g] -> FM3(B1){B1}G1[g]	36	40

M3(B1)[g] -> M3(B1)G1[g]	0	10
FM3(B1)[g] -> FM3(B1)G1[g]	0	10
M5(B1)[g] -> M5(B1)G1[g]	0	10
M4(B1)[g] -> M4(B1)G1[g]	0	10
FM5(B1)[g] -> FM5(B1)G1[g]	0	10
FM4(B1)[g] -> FM4(B1)G1[g]	0	10
M3(B1){B1}G1[g] -> M3(B1){B1}G2[g]	0	10
FM3(B1){B1}G1[g] -> FM3(B1){B1}G2[g]	9	10
M9[g] ->	0	10
M8[g] ->	0	10
M7[g] ->	0	10
M6[g] ->	0	10
M5[g] ->	0	10
M5(B1)[g] ->	0	10
M5(B1)G1[g] ->	0	10
M4(B1)[g] ->	0	10
M4(B1)G1[g] ->	0	10
M3(B1)[g] ->	0	10
M3(B1)G1[g] ->	0	10
M3(B1){B1}[g] ->	0	10
M3(B1){B1}G1[g] ->	0	10
M3(B1){B1}G2[g] ->	0	10
FM5(B1)[g] ->	0	10
FM5(B1)G1[g] ->	0	10
FM4(B1)[g] ->	0	10
FM4(B1)G1[g] ->	0	10
FM3(B1)[g] ->	0	10
FM3(B1)G1[g] ->	0	10
FM3(B1){B1}[g] ->	50	64
FM3(B1){B1}G1[g] ->	26	31
FM3(B1){B1}G2[g] ->	9	10

Table A.2: (Chapter 4.3.1) Model output after reducing the GnT1 ECP to zero.

Reaction	Min	Max
M9[g] ->	0	10
M8[g] ->	0	10
M7[g] ->	0	10
M6[g] ->	0	10
M5[g] ->	90	100

A.3 Chapter 4.3.2 Raw Output

Table A.3: (Chapter 4.3.2) Model output after calibration to control glycoform.

Reaction	Min	Max
-> M9[g]	100	100
M9[g] -> M8[g]	90	100
M8[g] -> M7[g]	90	100
M7[g] -> M6[g]	90	100
M6[g] -> M5[g]	90	100
M5[g] -> M5(B1)[g]	90	100
M5(B1)[g] -> M4(B1)[g]	90	100
M4(B1)[g] -> M3(B1)[g]	90	100
M3(B1)[g] -> M3(B1){B1}[g]	90	100
M3(B1){B1}[g] -> M3(B2)[g]	0	10
M5(B1)[g] -> M5(B2)[g]	0	10
M4(B1)[g] -> M4(B2)[g]	0	10
M3(B1){B1}[g] -> M3(B2){B1}[g]	54	60
M3(B1){B1}[g] -> M3(B1){B2}[g]	9	10
M3(B2){B1}[g] -> M3(B2){B2}[g]	9	10
M3(B1){B1}[g] -> FM3(B1){B1}[g]	27	37
M3(B2){B1}[g] -> FM3(B2){B1}[g]	45	50
M3(B1)[g] -> FM3(B1)[g]	0	10
M5(B1)[g] -> FM5(B1)[g]	0	10
M4(B1)[g] -> FM4(B1)[g]	0	10

M3(B1){B2}[g] -> FM3(B1){B2}[g]	9	10
M3(B2)[g] -> FM3(B2)[g]	0	10
M5(B2)[g] -> FM5(B2)[g]	0	10
M4(B2)[g] -> FM4(B2)[g]	0	10
M3(B2){B2}[g] -> FM3(B2){B2}[g]	9	10
M3(B1){B1}[g] -> M3(B1){B1}G1[g]	0	10
FM3(B1){B1}[g] -> FM3(B1){B1}G1[g]	27	37
M3(B2){B1}[g] -> M3(B2){B1}G1[g]	0	6
FM3(B2){B1}[g] -> FM3(B2){B1}G1[g]	45	50
M3(B1)[g] -> M3(B1)G1[g]	0	10
FM3(B1)[g] -> FM3(B1)G1[g]	0	10
M5(B1)[g] -> M5(B1)G1[g]	0	10
M4(B1)[g] -> M4(B1)G1[g]	0	10
M3(B1){B2}[g] -> M3(B1){B2}G1[g]	0	1
FM5(B1)[g] -> FM5(B1)G1[g]	0	10
FM4(B1)[g] -> FM4(B1)G1[g]	0	10
FM3(B1){B2}[g] -> FM3(B1){B2}G1[g]	9	10
M3(B2)[g] -> M3(B2)G1[g]	0	10
FM3(B2)[g] -> FM3(B2)G1[g]	0	10
M5(B2)[g] -> M5(B2)G1[g]	0	10
M4(B2)[g] -> M4(B2)G1[g]	0	10
M3(B2){B2}[g] -> M3(B2){B2}G1[g]	0	1
FM5(B2)[g] -> FM5(B2)G1[g]	0	10
FM4(B2)[g] -> FM4(B2)G1[g]	0	10
FM3(B2){B2}[g] -> FM3(B2){B2}G1[g]	9	10
M5(B2)G1[g] -> M5(B2)G2[g]	0	10
M4(B2)G1[g] -> M4(B2)G2[g]	0	10
M3(B2)G1[g] -> M3(B2)G2[g]	0	10
M3(B1){B1}G1[g] -> M3(B1){B1}G2[g]	0	10
M3(B2){B1}G1[g] -> M3(B2){B1}G2[g]	0	6
M3(B2){B2}G1[g] -> M3(B2){B2}G2[g]	0	1
FM5(B2)G1[g] -> FM5(B2)G2[g]	0	10
FM4(B2)G1[g] -> FM4(B2)G2[g]	0	10
FM4(B2)G1S1[g] -> FM4(B2)G2S1[g]	0	10
FM3(B2)G1[g] -> FM3(B2)G2[g]	0	10
FM3(B1){B1}G1[g] -> FM3(B1){B1}G2[g]	27	37
FM3(B1){B1}G1S1[g] -> FM3(B1){B1}G2S1[g]	0	10

FM3(B2){B1}G1[g] -> FM3(B2){B1}G2[g]	36	40
FM3(B2){B1}G1S1[g] -> FM3(B2){B1}G2S1[g]	0	14
FM3(B1){B2}G1[g] -> FM3(B1){B2}G2[g]	9	10
FM3(B2){B2}G1[g] -> FM3(B2){B2}G2[g]	9	10
M3(B2){B1}G2[g] -> M3(B2){B1}G3[g]	0	6
M3(B1){B2}G1[g] -> M3(B1){B2}G2[g]	0	1
M3(B1){B2}G2[g] -> M3(B1){B2}G3[g]	0	1
M3(B2){B2}G2[g] -> M3(B2){B2}G3[g]	0	1
FM3(B2){B1}G2[g] -> FM3(B2){B1}G3[g]	36	40
FM3(B1){B2}G2[g] -> FM3(B1){B2}G3[g]	9	10
FM3(B2){B2}G2[g] -> FM3(B2){B2}G3[g]	9	10
M3(B2){B2}G3[g] -> M3(B2){B2}G4[g]	0	1
FM3(B2){B2}G3[g] -> FM3(B2){B2}G4[g]	9	10
M5(B2)G1[g] -> M5(B2)G1S1[g]	0	10
M4(B2)G1[g] -> M4(B2)G1S1[g]	0	10
M3(B2)G1[g] -> M3(B2)G1S1[g]	0	10
M3(B1){B1}G1[g] -> M3(B1){B1}G1S1[g]	0	10
M3(B2){B1}G1[g] -> M3(B2){B1}G1S1[g]	0	6
M3(B2){B1}G2[g] -> M3(B2){B1}G2S1[g]	0	6
M3(B1){B2}G1[g] -> M3(B1){B2}G1S1[g]	0	1
M3(B1){B2}G2[g] -> M3(B1){B2}G2S1[g]	0	1
M3(B2){B2}G1[g] -> M3(B2){B2}G1S1[g]	0	1
M3(B2){B2}G2[g] -> M3(B2){B2}G2S1[g]	0	1
M3(B2){B2}G3[g] -> M3(B2){B2}G3S1[g]	0	1
FM5(B2)G1[g] -> FM5(B2)G1S1[g]	0	10
FM4(B2)G1[g] -> FM4(B2)G1S1[g]	0	10
FM3(B2)G1[g] -> FM3(B2)G1S1[g]	0	10

FM3(B1){B1}G1[g] -> FM3(B1){B1}G1S1[g]	0	10
FM3(B2){B1}G1[g] -> FM3(B2){B1}G1S1[g]	9	14
FM3(B2){B1}G2[g] -> FM3(B2){B1}G2S1[g]	0	4
FM3(B1){B2}G1[g] -> FM3(B1){B2}G1S1[g]	0	1
FM3(B1){B2}G2[g] -> FM3(B1){B2}G2S1[g]	0	1
FM3(B2){B2}G1[g] -> FM3(B2){B2}G1S1[g]	0	1
FM3(B2){B2}G2[g] -> FM3(B2){B2}G2S1[g]	0	1
FM3(B2){B2}G3[g] -> FM3(B2){B2}G3S1[g]	0	1
M5(B1)G1[g] -> M5(B1)G1S1[g]	0	10
M5(B2)G2[g] -> M5(B2)G2S1[g]	0	10
M4(B1)G1[g] -> M4(B1)G1S1[g]	0	10
M4(B2)G2[g] -> M4(B2)G2S1[g]	0	10
M3(B1)G1[g] -> M3(B1)G1S1[g]	0	10
M3(B2)G2[g] -> M3(B2)G2S1[g]	0	10
M3(B1){B1}G2[g] -> M3(B1){B1}G2S1[g]	0	10
M3(B2){B1}G3[g] -> M3(B2){B1}G3S1[g]	0	6
M3(B1){B2}G3[g] -> M3(B1){B2}G3S1[g]	0	1
M3(B2){B2}G4[g] -> M3(B2){B2}G4S1[g]	0	1
FM5(B1)G1[g] -> FM5(B1)G1S1[g]	0	10
FM5(B2)G2[g] -> FM5(B2)G2S1[g]	0	10
FM4(B1)G1[g] -> FM4(B1)G1S1[g]	0	10
FM4(B2)G2[g] -> FM4(B2)G2S1[g]	0	10
FM3(B1)G1[g] -> FM3(B1)G1S1[g]	0	10
FM3(B2)G2[g] -> FM3(B2)G2S1[g]	0	10
FM3(B1){B1}G2[g] -> FM3(B1){B1}G2S1[g]	18	20
FM3(B2){B1}G3[g] -> FM3(B2){B1}G3S1[g]	18	20
FM3(B1){B2}G3[g] -> FM3(B1){B2}G3S1[g]	9	10
FM3(B2){B2}G4[g] -> FM3(B2){B2}G4S1[g]	9	10

M3(B2){B1}G2S1[g] -> M3(B2){B1}G2S2[g]	0	6
M3(B1){B2}G2S1[g] -> M3(B1){B2}G2S2[g]	0	1
M3(B2){B2}G2S1[g] -> M3(B2){B2}G2S2[g]	0	1
M3(B2){B2}G3S1[g] -> M3(B2){B2}G3S2[g]	0	1
FM3(B2){B1}G2S1[g] -> FM3(B2){B1}G2S2[g]	0	10
FM3(B1){B2}G2S1[g] -> FM3(B1){B2}G2S2[g]	0	1
FM3(B2){B2}G2S1[g] -> FM3(B2){B2}G2S2[g]	0	1
FM3(B2){B2}G3S1[g] -> FM3(B2){B2}G3S2[g]	0	1
M5(B2)G2S1[g] -> M5(B2)G2S2[g]	0	10
M4(B2)G2S1[g] -> M4(B2)G2S2[g]	0	10
M3(B2)G2S1[g] -> M3(B2)G2S2[g]	0	10
M3(B1){B1}G2S1[g] -> M3(B1){B1}G2S2[g]	0	10
M3(B2){B1}G3S1[g] -> M3(B2){B1}G3S2[g]	0	6
M3(B1){B2}G3S1[g] -> M3(B1){B2}G3S2[g]	0	1
M3(B2){B2}G4S1[g] -> M3(B2){B2}G4S2[g]	0	1
FM5(B2)G2S1[g] -> FM5(B2)G2S2[g]	0	10
FM4(B2)G2S1[g] -> FM4(B2)G2S2[g]	0	10
FM3(B2)G2S1[g] -> FM3(B2)G2S2[g]	0	10
FM3(B1){B1}G2S1[g] -> FM3(B1){B1}G2S2[g]	9	10
FM3(B2){B1}G3S1[g] -> FM3(B2){B1}G3S2[g]	9	10
FM3(B1){B2}G3S1[g] -> FM3(B1){B2}G3S2[g]	9	10
FM3(B2){B2}G4S1[g] -> FM3(B2){B2}G4S2[g]	9	10
M9[g] ->	0	10
M8[g] ->	0	10
M7[g] ->	0	10
M6[g] ->	0	10
M5[g] ->	0	10
M5(B1)[g] ->	0	10

M5(B1)G1[g] ->	0	10
M5(B1)G1S1[g] ->	0	10
M5(B2)[g] ->	0	10
M5(B2)G1[g] ->	0	10
M5(B2)G1S1[g] ->	0	10
M5(B2)G2[g] ->	0	10
M5(B2)G2S1[g] ->	0	10
M5(B2)G2S2[g] ->	0	10
M4(B1)[g] ->	0	10
M4(B1)G1[g] ->	0	10
M4(B1)G1S1[g] ->	0	10
M4(B2)[g] ->	0	10
M4(B2)G1[g] ->	0	10
M4(B2)G1S1[g] ->	0	10
M4(B2)G2[g] ->	0	10
M4(B2)G2S1[g] ->	0	10
M4(B2)G2S2[g] ->	0	10
M3(B1)[g] ->	0	10
M3(B1)G1[g] ->	0	10
M3(B1)G1S1[g] ->	0	10
M3(B2)[g] ->	0	10
M3(B2)G1[g] ->	0	10
M3(B2)G1S1[g] ->	0	10
M3(B2)G2[g] ->	0	10
M3(B2)G2S1[g] ->	0	10
M3(B2)G2S2[g] ->	0	10
M3(B1){B1}[g] ->	0	10
M3(B1){B1}G1[g] ->	0	10
M3(B1){B1}G1S1[g] ->	0	10
M3(B1){B1}G2[g] ->	0	10
M3(B1){B1}G2S1[g] ->	0	10
M3(B1){B1}G2S2[g] ->	0	10
M3(B2){B1}[g] ->	0	6
M3(B2){B1}G1[g] ->	0	6
M3(B2){B1}G1S1[g] ->	0	6
M3(B2){B1}G2[g] ->	0	6
M3(B2){B1}G2S1[g] ->	0	6
M3(B2){B1}G2S2[g] ->	0	6
M3(B2){B1}G3[g] ->	0	6

M3(B2){B1}G3S1[g] ->	0	6
M3(B2){B1}G3S2[g] ->	0	6
M3(B1){B2}[g] ->	0	1
M3(B1){B2}G1[g] ->	0	1
M3(B1){B2}G1S1[g] ->	0	1
M3(B1){B2}G2[g] ->	0	1
M3(B1){B2}G2S1[g] ->	0	1
M3(B1){B2}G2S2[g] ->	0	1
M3(B1){B2}G3[g] ->	0	1
M3(B1){B2}G3S1[g] ->	0	1
M3(B1){B2}G3S2[g] ->	0	1
M3(B2){B2}[g] ->	0	1
M3(B2){B2}G1[g] ->	0	1
M3(B2){B2}G1S1[g] ->	0	1
M3(B2){B2}G2[g] ->	0	1
M3(B2){B2}G2S1[g] ->	0	1
M3(B2){B2}G2S2[g] ->	0	1
M3(B2){B2}G3[g] ->	0	1
M3(B2){B2}G3S1[g] ->	0	1
M3(B2){B2}G3S2[g] ->	0	1
M3(B2){B2}G4[g] ->	0	1
M3(B2){B2}G4S1[g] ->	0	1
M3(B2){B2}G4S2[g] ->	0	1
FM5(B1)[g] ->	0	10
FM5(B1)G1[g] ->	0	10
FM5(B1)G1S1[g] ->	0	10
FM5(B2)[g] ->	0	10
FM5(B2)G1[g] ->	0	10
FM5(B2)G1S1[g] ->	0	10
FM5(B2)G2[g] ->	0	10
FM5(B2)G2S1[g] ->	0	10
FM5(B2)G2S2[g] ->	0	10
FM4(B1)[g] ->	0	10
FM4(B1)G1[g] ->	0	10
FM4(B1)G1S1[g] ->	0	10
FM4(B2)[g] ->	0	10
FM4(B2)G1[g] ->	0	10
FM4(B2)G1S1[g] ->	0	10
FM4(B2)G2[g] ->	0	10

FM4(B2)G2S1[g] ->	0	10
FM4(B2)G2S2[g] ->	0	10
FM3(B1)[g] ->	0	10
FM3(B1)G1[g] ->	0	10
FM3(B1)G1S1[g] ->	0	10
FM3(B2)[g] ->	0	10
FM3(B2)G1[g] ->	0	10
FM3(B2)G1S1[g] ->	0	10
FM3(B2)G2[g] ->	0	10
FM3(B2)G2S1[g] ->	0	10
FM3(B2)G2S2[g] ->	0	10
FM3(B1){B1}[g] ->	0	10
FM3(B1){B1}G1[g] ->	0	10
FM3(B1){B1}G1S1[g] ->	0	10
FM3(B1){B1}G2[g] ->	7	19
FM3(B1){B1}G2S1[g] ->	8	21
FM3(B1){B1}G2S2[g] ->	9	10
FM3(B2){B1}[g] ->	0	5
FM3(B2){B1}G1[g] ->	0	5
FM3(B2){B1}G1S1[g] ->	0	14
FM3(B2){B1}G2[g] ->	0	4
FM3(B2){B1}G2S1[g] ->	0	14
FM3(B2){B1}G2S2[g] ->	0	10
FM3(B2){B1}G3[g] ->	16	22
FM3(B2){B1}G3S1[g] ->	8	11
FM3(B2){B1}G3S2[g] ->	9	10
FM3(B1){B2}[g] ->	0	1
FM3(B1){B2}G1[g] ->	0	1
FM3(B1){B2}G1S1[g] ->	0	1
FM3(B1){B2}G2[g] ->	0	1
FM3(B1){B2}G2S1[g] ->	0	1
FM3(B1){B2}G2S2[g] ->	0	1
FM3(B1){B2}G3[g] ->	0	1
FM3(B1){B2}G3S1[g] ->	0	1
FM3(B1){B2}G3S2[g] ->	9	10
FM3(B2){B2}[g] ->	0	1
FM3(B2){B2}G1[g] ->	0	1
FM3(B2){B2}G1S1[g] ->	0	1
FM3(B2){B2}G2[g] ->	0	1

FM3(B2){B2}G2S1[g] ->	0	1
FM3(B2){B2}G2S2[g] ->	0	1
FM3(B2){B2}G3[g] ->	0	1
FM3(B2){B2}G3S1[g] ->	0	1
FM3(B2){B2}G3S2[g] ->	0	1
FM3(B2){B2}G4[g] ->	0	1
FM3(B2){B2}G4S1[g] ->	0	1
FM3(B2){B2}G4S2[g] ->	9	10

Table A.4: (Chapter 4.3.2) Model output after reducing the GnTI ECP to zero.

Reaction	Min	Max
M9[g] ->	0	10
M8[g] ->	0	10
M7[g] ->	0	10
M6[g] ->	0	10
M5[g] ->	90	100

A.4 Chapter 4.3.3 Raw Output

Table A.5: (Chapter 4.3.3) Model output after calibration to experimental glycoform.

Reaction	Min	Max
-> M9[g]	100	100
M9[g] -> M8[g]	90	100
M8[g] -> M7[g]	90	100
M7[g] -> M6[g]	90	100
M6[g] -> M5[g]	90	100
M5[g] -> M5(B1)[g]	90	100
M5(B1)[g] -> M4(B1)[g]	90	100
M4(B1)[g] -> M3(B1)[g]	90	100
M3(B1)[g] -> M3(B1){B1}[g]	90	100

M3(B1){B1}[g] -> FM3(B1){B1}[g]	90	100
M3(B1)[g] -> FM3(B1)[g]	0	10
M5(B1)[g] -> FM5(B1)[g]	0	10
M4(B1)[g] -> FM4(B1)[g]	0	10
M3(B1){B1}[g] -> M3(B1){B1}G1[g]	0	10
FM3(B1){B1}[g] -> FM3(B1){B1}G1[g]	36	40
M3(B1)[g] -> M3(B1)G1[g]	0	10
FM3(B1)[g] -> FM3(B1)G1[g]	0	10
M5(B1)[g] -> M5(B1)G1[g]	0	10
M4(B1)[g] -> M4(B1)G1[g]	0	10
FM5(B1)[g] -> FM5(B1)G1[g]	0	10
FM4(B1)[g] -> FM4(B1)G1[g]	0	10
M3(B1){B1}G1[g] -> M3(B1){B1}G2[g]	0	10
FM3(B1){B1}G1[g] -> FM3(B1){B1}G2[g]	9	10
M9[g] ->	0	10
M8[g] ->	0	10
M7[g] ->	0	10
M6[g] ->	0	10
M5[g] ->	0	10
M5(B1)[g] ->	0	10
M5(B1)G1[g] ->	0	10
M4(B1)[g] ->	0	10
M4(B1)G1[g] ->	0	10
M3(B1)[g] ->	0	10
M3(B1)G1[g] ->	0	10
M3(B1){B1}[g] ->	0	10
M3(B1){B1}G1[g] ->	0	10
M3(B1){B1}G2[g] ->	0	10
FM5(B1)[g] ->	0	10
FM5(B1)G1[g] ->	0	10
FM4(B1)[g] ->	0	10
FM4(B1)G1[g] ->	0	10
FM3(B1)[g] ->	0	10
FM3(B1)G1[g] ->	0	10
FM3(B1){B1}[g] ->	50	64
FM3(B1){B1}G1[g] ->	26	31
FM3(B1){B1}G2[g] ->	9	10

Table A.6: (Chapter 4.3.3) Model output after reducing FucT ECP to zero.

Reaction	Min	Max
-> M9[g]	100	100
M9[g] -> M8[g]	90	100
M8[g] -> M7[g]	90	100
M7[g] -> M6[g]	90	100
M6[g] -> M5[g]	90	100
M5[g] -> M5(B1)[g]	90	100
M5(B1)[g] -> M4(B1)[g]	90	100
M4(B1)[g] -> M3(B1)[g]	90	100
M3(B1)[g] -> M3(B1){B1}[g]	90	100
M3(B1){B1}[g] -> M3(B1){B1}G1[g]	36	40
M3(B1)[g] -> M3(B1)G1[g]	0	10
M5(B1)[g] -> M5(B1)G1[g]	0	10
M4(B1)[g] -> M4(B1)G1[g]	0	10
M3(B1){B1}G1[g] -> M3(B1){B1}G2[g]	9	10
M9[g] ->	0	10
M8[g] ->	0	10
M7[g] ->	0	10
M6[g] ->	0	10
M5[g] ->	0	10
M5(B1)[g] ->	0	10
M5(B1)G1[g] ->	0	10
M4(B1)[g] ->	0	10
M4(B1)G1[g] ->	0	10
M3(B1)[g] ->	0	10
M3(B1)G1[g] ->	0	10
M3(B1){B1}[g] ->	50	64
M3(B1){B1}G1[g] ->	26	31
M3(B1){B1}G2[g] ->	9	10

A.5 Chapter 4.3.4 Raw Output

Table A.7: (Chapter 4.3.4) Model output after calibration to control glycoform.

Reaction	Min	Max
-> M9[g]	100	100
M9[g] -> M8[g]	90	100
M8[g] -> M7[g]	90	100
M7[g] -> M6[g]	90	100
M6[g] -> M5[g]	90	100
M5[g] -> M5(B1)[g]	90	100
M5(B1)[g] -> M4(B1)[g]	90	100
M4(B1)[g] -> M3(B1)[g]	90	100
M3(B1)[g] -> M3(B1){B1}[g]	90	100
M3(B1){B1}[g] -> FM3(B1){B1}[g]	90	100
M3(B1)[g] -> FM3(B1)[g]	0	10
M5(B1)[g] -> FM5(B1)[g]	0	10
M4(B1)[g] -> FM4(B1)[g]	0	10
M3(B1){B1}[g] -> M3(B1){B1}G1[g]	0	10
FM3(B1){B1}[g] -> FM3(B1){B1}G1[g]	54	60
M3(B1)[g] -> M3(B1)G1[g]	0	10
FM3(B1)[g] -> FM3(B1)G1[g]	0	10
M5(B1)[g] -> M5(B1)G1[g]	0	10
M4(B1)[g] -> M4(B1)G1[g]	0	10
FM5(B1)[g] -> FM5(B1)G1[g]	0	10
FM4(B1)[g] -> FM4(B1)G1[g]	0	10
M3(B1){B1}G1[g] -> M3(B1){B1}G2[g]	0	10
FM3(B1){B1}G1[g] -> FM3(B1){B1}G2[g]	18	20
M9[g] ->	0	10
M8[g] ->	0	10
M7[g] ->	0	10

M6[g] ->	0	10
M5[g] ->	0	10
M5(B1)[g] ->	0	10
M5(B1)G1[g] ->	0	10
M4(B1)[g] ->	0	10
M4(B1)G1[g] ->	0	10
M3(B1)[g] ->	0	10
M3(B1)G1[g] ->	0	10
M3(B1){B1}[g] ->	0	10
M3(B1){B1}G1[g] ->	0	10
M3(B1){B1}G2[g] ->	0	10
FM5(B1)[g] ->	0	10
FM5(B1)G1[g] ->	0	10
FM4(B1)[g] ->	0	10
FM4(B1)G1[g] ->	0	10
FM3(B1)[g] ->	0	10
FM3(B1)G1[g] ->	0	10
FM3(B1){B1}[g] ->	30	46
FM3(B1){B1}G1[g] ->	34	42
FM3(B1){B1}G2[g] ->	18	20

Table A.8: (Chapter 4.3.4) Model output after calibration to experimental glycoform (*ST6Gal1 overexpression*).

Reaction	Min	Max
-> M9[g]	100	100
M9[g] -> M8[g]	90	100
M8[g] -> M7[g]	90	100
M7[g] -> M6[g]	90	100
M6[g] -> M5[g]	90	100
M5[g] -> M5(B1)[g]	90	100

M5(B1)[g] -> M4(B1)[g]	90	100
M4(B1)[g] -> M3(B1)[g]	90	100
M3(B1)[g] -> M3(B1){B1}[g]	90	100
M3(B1){B1}[g] -> FM3(B1){B1}[g]	72	80
M3(B1)[g] -> FM3(B1)[g]	0	10
M5(B1)[g] -> FM5(B1)[g]	0	10
M4(B1)[g] -> FM4(B1)[g]	0	10
M3(B1){B1}[g] -> M3(B1){B1}G1[g]	18	28
FM3(B1){B1}[g] -> FM3(B1){B1}G1[g]	54	60
M3(B1)[g] -> M3(B1)G1[g]	0	10
FM3(B1)[g] -> FM3(B1)G1[g]	0	10
M5(B1)[g] -> M5(B1)G1[g]	0	10
M4(B1)[g] -> M4(B1)G1[g]	0	10
FM5(B1)[g] -> FM5(B1)G1[g]	0	10
FM4(B1)[g] -> FM4(B1)G1[g]	0	10
M3(B1){B1}G1[g] -> M3(B1){B1}G2[g]	18	28
FM3(B1){B1}G1[g] -> FM3(B1){B1}G2[g]	54	60
FM3(B1){B1}G1S1[g] -> FM3(B1){B1}G2S1[g]	0	6
M3(B1){B1}G1[g] -> M3(B1){B1}G1S1[g]	0	10
FM3(B1){B1}G1[g] -> FM3(B1){B1}G1S1[g]	0	6
M5(B1)G1[g] -> M5(B1)G1S1[g]	0	10
M4(B1)G1[g] -> M4(B1)G1S1[g]	0	10
M3(B1)G1[g] -> M3(B1)G1S1[g]	0	10
M3(B1){B1}G2[g] -> M3(B1){B1}G2S1[g]	18	28
FM5(B1)G1[g] -> FM5(B1)G1S1[g]	0	10
FM4(B1)G1[g] -> FM4(B1)G1S1[g]	0	10
FM3(B1)G1[g] -> FM3(B1)G1S1[g]	0	10
FM3(B1){B1}G2[g] -> FM3(B1){B1}G2S1[g]	45	50
M3(B1){B1}G2S1[g] -> M3(B1){B1}G2S2[g]	18	20
FM3(B1){B1}G2S1[g] -> FM3(B1){B1}G2S2[g]	18	20
M9[g] ->	0	10

M8[g] ->	0	10
M7[g] ->	0	10
M6[g] ->	0	10
M5[g] ->	0	10
M5(B1)[g] ->	0	10
M5(B1)G1[g] ->	0	10
M5(B1)G1S1[g] ->	0	10
M4(B1)[g] ->	0	10
M4(B1)G1[g] ->	0	10
M4(B1)G1S1[g] ->	0	10
M3(B1)[g] ->	0	10
M3(B1)G1[g] ->	0	10
M3(B1)G1S1[g] ->	0	10
M3(B1){B1}[g] ->	0	10
M3(B1){B1}G1[g] ->	0	10
M3(B1){B1}G1S1[g] ->	0	10
M3(B1){B1}G2[g] ->	0	10
M3(B1){B1}G2S1[g] ->	0	10
M3(B1){B1}G2S2[g] ->	18	20
FM5(B1)[g] ->	0	10
FM5(B1)G1[g] ->	0	10
FM5(B1)G1S1[g] ->	0	10
FM4(B1)[g] ->	0	10
FM4(B1)G1[g] ->	0	10
FM4(B1)G1S1[g] ->	0	10
FM3(B1)[g] ->	0	10
FM3(B1)G1[g] ->	0	10
FM3(B1)G1S1[g] ->	0	10
FM3(B1){B1}[g] ->	12	26
FM3(B1){B1}G1[g] ->	0	6
FM3(B1){B1}G1S1[g] ->	0	6
FM3(B1){B1}G2[g] ->	4	15
FM3(B1){B1}G2S1[g] ->	25	38
FM3(B1){B1}G2S2[g] ->	18	20

Appendix B

RAW MODEL OUTPUT (CHAPTER 5 RESULTS)

B.1 Preface

This chapter provides the ECPs obtained for all experiment analyzed in Chapter 5.

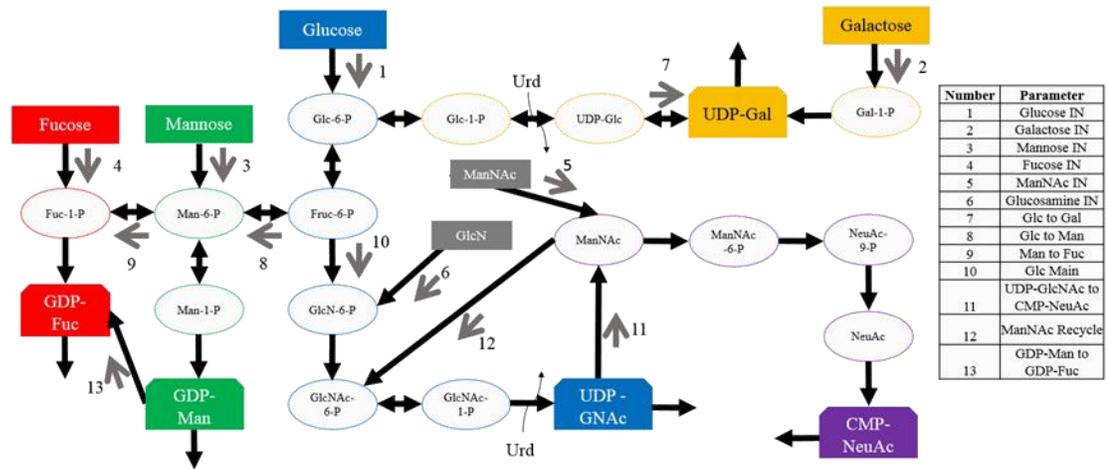


Figure B.1: Reactions corresponding to all ECP values.

B.2 Chapter 5.3.1 ECP Values

Table B.1: (Chapter 5.3.1) ECP values for all experimental conditions.

Parameter	Control	Gal	Gal + Urd	N-acetylman.	N-acetylman.+ cytidine	Glucosamine	Glucosamine + Urd
Glucose IN	1294.5	1188.7	1199.2	1123.3	1122.5	1081.1	1080.0
Galactose IN	0.0	167.6	166.4	0.0	0.0	0.0	0.0
Mannose IN	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Fucose IN	0.0	0.0	0.0	0.0	0.0	0.0	0.0
ManNAc IN	0.0	0.0	0.0	303.6	355.8	0.0	0.0
Glucosamine IN	0.0	0.0	0.0	0.0	0.0	274.8	273.3
Glc to Gal	152.7	0.0	0.0	183.6	191.2	153.5	149.2
Glc to Man	929.3	930.5	926.1	939.7	931.3	927.6	930.8
Man to Fuc	29.3	30.5	26.1	39.7	31.3	27.6	30.8
Glc Main	212.5	258.2	273.1	0.0	0.0	0.0	0.0
UDP-GlcNAc to CMP-NeuAc	44.4	76.8	67.7	112.7	134.6	104.7	114.0
ManNAc Recycle	0.0	0.0	0.0	190.8	221.2	0.0	0.0
GDP-Man to GDP-Fuc	0.0	0.0	0.0	0.0	0.0	0.0	0.0

B.3 Chapter 5.3.2 ECP Values

Table B.2: (Chapter 5.3.2) ECP values for all experimental conditions.

Parameter	Control	0x UMG Suboptimal	0x UMG	1x UMG	2x UMG	3x UMG	4x UMG	8x UMG	12x UMG	16x UMG	20x UMG
Glucose IN	1231.7	1216.7	1199.3	1195.2	1190.5	1190.7	1190.2	1189.3	1189.8	1189.8	1185.9
Galactose IN	0.0	0.0	0.0	20.3	26.8	31.0	36.4	43.0	43.5	46.6	46.1
Mannose IN	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Fucose IN	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
ManNAc IN	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Glucosamine IN	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Glc to Gal	36.1	19.1	5.8	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Glc to Man	996.9	998.0	995.7	996.7	993.8	994.0	993.7	993.0	993.1	993.3	990.5
Man to Fuc	96.9	98.0	95.7	96.7	93.8	94.0	93.7	93.0	93.1	93.3	90.5
Glc Main	198.8	199.6	197.8	198.6	196.7	196.7	196.5	196.3	196.7	196.5	195.4
UDP-GlcNAc to CMP-NeuAc	0.6	0.4	0.2	0.4	1.4	1.2	1.4	1.0	1.2	1.6	1.7
ManNAc Recycle	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
GDP-Man to GDP-Fuc	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

B.4 Chapter 5.3.3 ECP Values

Table B.3: (Chapter 5.3.3) ECP values for all experimental conditions.

Parameter	Control	Mn	Gal	NH4	Mn + NH4	Mn + Gal	Gal + NH4	Mn + Gal + NH4
Glucose IN	1182.2	1119.7	1143.8	1169.3	1083.1	1077.6	1126.6	1028.1
Galactose IN	0.0	0.0	49.7	0.0	0.0	35.2	35.8	20.5
Mannose IN	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Fucose IN	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
ManNAc IN	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Glucosamine IN	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Glc to Gal	33.0	27.4	0.0	28.0	18.0	0.0	0.0	0.0
Glc to Man	972.5	944.0	969.5	968.8	933.9	942.4	966.6	928.9
Man to Fuc	72.5	44.0	69.5	68.8	33.9	42.4	66.6	28.9
Glc Main	176.7	148.3	174.3	172.4	131.2	135.3	160.0	99.2
UDP-GlcNAc to CMP-NeuAc	1.0	1.0	1.3	1.3	1.1	0.8	0.5	0.5
MANNAc Recycle	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
GDP-Man to GDP-Fuc	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

B.5 Chapter 5.3.4 ECP Values

Table B.4: (Chapter 5.3.4) ECP values for all experimental conditions.

Gln Feeding Concentration	0 mM			4 mM			8 mM		
	Day 1	Day 3	Day 5	Day 1	Day 3	Day 5	Day 1	Day 3	Day 5
Glucose IN	1318.3	1298.8	1321.0	1421.5	1387.8	1384.4	1422.5	1379.0	1381.2
Galactose IN	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Mannose IN	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Fucose IN	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
ManNAc IN	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Glucosamine IN	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Glc to Gal	138.8	130.9	142.0	174.3	162.8	160.4	175.7	159.3	158.9
Glc to Man	933.9	932.0	934.5	942.7	939.8	939.9	943.1	939.1	939.3
Man to Fuc	33.9	32.0	34.5	42.7	39.8	39.9	43.1	39.1	39.3
Glc Main	245.6	235.9	244.5	304.5	285.2	284.2	303.7	280.7	283.0
UDP-GlcNAc to CMP-NeuAc	72.3	67.2	70.0	103.0	91.3	91.5	102.1	90.1	91.1
MAnNAc Recycle	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
GDP-Man to GDP-Fuc	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Appendix C

MATLAB SOURCE CODE

C.1 Preface

This section contains all source code used in the model.

C.2 Source Code

C.2.1 GUI Interface (File – GlycoMapperExec.m)

```
% THIS SCRIPT CONTROLS THE GUI FOR THE MODEL

function varargout = GlycoMapperExec(varargin)
% GLYCOMAPPEREXEC MATLAB code for GlycoMapperExec.fig
%     GLYCOMAPPEREXEC, by itself, creates a new
GLYCOMAPPEREXEC or raises the existing
%     singleton*.
%
%     H = GLYCOMAPPEREXEC returns the handle to a new
GLYCOMAPPEREXEC or the handle to
%     the existing singleton*.
%
%
GLYCOMAPPEREXEC('CALLBACK',hObject,eventData,handles,...) calls
the local
%     function named CALLBACK in GLYCOMAPPEREXEC.M with the
given input arguments.
%
%     GLYCOMAPPEREXEC('Property','Value',...) creates a new
GLYCOMAPPEREXEC or raises the
%     existing singleton*. Starting from the left, property
value pairs are
%     applied to the GUI before GlycoMapperExec_OpeningFcn
gets called. An
%     unrecognized property name or invalid value makes
property application
%     stop. All inputs are passed to
GlycoMapperExec_OpeningFcn via varargin.
```

```

%
%      *See GUI Options on GUIDE's Tools menu.  Choose "GUI
allows only one
%      instance to run (singleton)".
%
% See also: GUIDE, GUIDATA, GUIHANDLES

% Edit the above text to modify the response to help
GlycoMapperExec

% Last Modified by GUIDE v2.5 29-Jan-2018 13:28:46

% Begin initialization code - DO NOT EDIT
gui_Singleton = 1;
gui_State = struct('gui_Name',       mfilename, ...
                  'gui_Singleton',  gui_Singleton, ...
                  'gui_OpeningFcn', @GlycoMapperExec_OpeningFcn, ...
                  'gui_OutputFcn',  @GlycoMapperExec_OutputFcn, ...
                  'gui_LayoutFcn',  [] , ...
                  'gui_Callback',   []);
if nargin && ischar(varargin{1})
    gui_State.gui_Callback = str2func(varargin{1});
end

if nargout
    [varargout{1:nargout}] = gui_mainfcn(gui_State,
varargin{:});
else
    gui_mainfcn(gui_State, varargin{:});
end
% End initialization code - DO NOT EDIT

% --- Executes just before GlycoMapperExec is made visible.
function GlycoMapperExec_OpeningFcn(hObject, eventdata,
handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to GlycoMapperExec (see
VARARGIN)

% Choose default command line output for GlycoMapperExec
handles.output = hObject;

```

```

% Update handles structure
guidata(hObject, handles);

[junk1, junk2, data] =
xlsread('GlycoSheet_Global_CRN.xlsx','Sheet Setup','A1:B18');
set(handles.Results_Table,'Data',data);

% UIWAIT makes GlycoMapperExec wait for user response (see
UIRESUME)
% uiwait(handles.figure1);

% --- Outputs from this function are returned to the command
line.
function varargout = GlycoMapperExec_OutputFcn(hObject,
eventdata, handles)
% varargout cell array for returning output args (see
VARARGOUT);
% hObject handle to figure
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Get default command line output from handles structure
varargout{1} = handles.output;

% --- Executes on button press in Exploration_Button.
function Exploration_Button_Callback(hObject, eventdata,
handles)
% hObject handle to Exploration_Button (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

%First we have to make sure the button grabs data from the
parameters
%section of the excel sheet, otherwise it will read in garbage
values

if get(handles.view_Params,'Value') ~= 1

    set(handles.view_Params,'Value',1);

```

```

        refresh_Callback(hObject, eventdata, handles);
end

input = get(handles.Results_Table, 'Data');

%Now we can actually compute and solve the problem.

data = MAIN_CRN_ExplorationMode(input);
set(handles.Results_Table, 'ColumnEditable', false(1,3));

set(handles.Results_Table, 'Data', data);
set(handles.view_Results, 'Value', 1);

% --- Executes on button press in Mapping_Button_Algl.
function Mapping_Button_Algl_Callback(hObject, eventdata,
handles)
% hObject    handle to Mapping_Button_Algl (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

if get(handles.view_Glycans, 'Value') ~= 1

    set(handles.view_Glycans, 'Value', 1);

    refresh_Callback(hObject, eventdata, handles);
end
input = get(handles.Results_Table, 'Data');

if get(handles.Enzyme_Mode, 'Value') == 1
    data = MAIN_CRN_MappingModeAlg1(input);
else
    objectiveArray = zeros(5,2);

    if get(handles.galFeed_Checkbox, 'Value') == 1
        objectiveArray(1,:) = [0 100000];
    end
    if get(handles.manFeed_Checkbox, 'Value') == 1
        objectiveArray(2,:) = [0 100000];
    end
    if get(handles.fucFeed_Checkbox, 'Value') == 1

```

```

        objectiveArray(3,:) = [0 100000];
    end
    if get(handles.manNACFeed_Checkbox, 'Value') == 1
        objectiveArray(4,:) = [0 100000];
    end
    if get(handles.glcNFeed_Checkbox, 'Value') == 1
        objectiveArray(5,:) = [0 100000];
    end

    data = MAIN_CCM_MappingModeAlg1(input, objectiveArray);
    set(handles.Results_Table, 'ColumnEditable', false(1,2));
end

set(handles.Results_Table, 'Data', data);
set(handles.view_Params, 'Value', 1);

% --- Executes on button press in Mapping_Button_Alq2.
function Mapping_Button_Alq2_Callback(hObject, eventdata, handles)
% hObject    handle to Mapping_Button_Alq2 (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

if get(handles.view_Glycans, 'Value') ~= 1

    set(handles.view_Glycans, 'Value', 1);

    refresh_Callback(hObject, eventdata, handles);
end
input = get(handles.Results_Table, 'Data');

data = MAIN_CRN_MappingModeAlg2(input);

set(handles.Results_Table, 'Data', data);
set(handles.view_Params, 'Value', 1);

% --- Executes on button press in refresh.
function refresh_Callback(hObject, eventdata, handles)
% hObject    handle to refresh (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

```

```

if get(handles.Enzyme_Mode, 'Value') == 1

    if get(handles.view_Params, 'Value') == 1
        [junk1, junk2, data] =
xlsread('GlycoSheet_Global_CRN.xlsx', 'Sheet Setup', 'A1:B18');
        set(handles.Results_Table, 'ColumnEditable', [false
true]);
    else
        if get(handles.view_Results, 'Value') == 1
            [junk1, junk2, data] =
xlsread('GlycoSheet_Global_CRN.xlsx', 'Results', 'A1:C500');

set(handles.Results_Table, 'ColumnEditable', false(1,3));
        else
            [junk1, junk2, data] =
xlsread('GlycoSheet_Global_CRN.xlsx', 'Sheet Setup', 'D1:E27');
            set(handles.Results_Table, 'ColumnEditable', [true
true]);
        end
    end
else
    if get(handles.view_Params, 'Value') == 1
        [junk1, junk2, data] =
xlsread('GlycoSheet_Global_CCM.xlsx', 'Sheet Setup', 'A1:B14');
        set(handles.Results_Table, 'ColumnEditable', false(1,2));
    else
        if get(handles.view_Results, 'Value') == 1
            [junk1, junk2, data] =
xlsread('GlycoSheet_Global_CCM.xlsx', 'Results', 'A1:B50');

set(handles.Results_Table, 'ColumnEditable', false(1,2));
        else
            [junk1, junk2, data] =
xlsread('GlycoSheet_Global_CCM.xlsx', 'Sheet Setup', 'D1:E27');
            set(handles.Results_Table, 'ColumnEditable', [true
true]);
        end
    end
end

set(handles.Results_Table, 'Data', data);

% --- Executes on button press in Metabolism_Mode.
function Metabolism_Mode_Callback(hObject, eventdata, handles)
% hObject    handle to Metabolism_Mode (see GCBO)

```

```

% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of
Metabolism_Mode

set(handles.Exploration_Button,'enable','off');
set(handles.Exploration_Button,'visible','off');

set(handles.Mapping_Button_Alg2,'enable','off');
set(handles.Mapping_Button_Alg2,'visible','off');
set(handles.mappingOptionsPanel,'visible','on');

% --- Executes on button press in Enzyme_Mode.
function Enzyme_Mode_Callback(hObject, eventdata, handles)
% hObject handle to Enzyme_Mode (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of
Enzyme_Mode

set(handles.Exploration_Button,'enable','on');
set(handles.Exploration_Button,'visible','on');

set(handles.Mapping_Button_Alg2,'enable','on');
set(handles.Mapping_Button_Alg2,'visible','on');
set(handles.mappingOptionsPanel,'visible','off');

% --- Executes on button press in manFeed_Checkbox.
function manFeed_Checkbox_Callback(hObject, eventdata, handles)
% hObject handle to manFeed_Checkbox (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of
manFeed_Checkbox

% --- Executes on button press in fucFeed_Checkbox.
function fucFeed_Checkbox_Callback(hObject, eventdata, handles)
% hObject handle to fucFeed_Checkbox (see GCBO)

```

```

% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of
fucFeed_Checkbox

% --- Executes on button press in galFeed_Checkbox.
function galFeed_Checkbox_Callback(hObject, eventdata, handles)
% hObject handle to galFeed_Checkbox (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of
galFeed_Checkbox

% --- Executes on button press in glcNFeed_Checkbox.
function glcNFeed_Checkbox_Callback(hObject, eventdata,
handles)
% hObject handle to glcNFeed_Checkbox (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of
glcNFeed_Checkbox

% --- Executes on button press in manNacFeed_Checkbox.
function manNacFeed_Checkbox_Callback(hObject, eventdata,
handles)
% hObject handle to manNacFeed_Checkbox (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB
% handles structure with handles and user data (see GUIDATA)

% Hint: get(hObject,'Value') returns toggle state of
manNacFeed_Checkbox

% --- Executes on button press in Paste.
function Paste_Callback(hObject, eventdata, handles)
% hObject handle to Paste (see GCBO)
% eventdata reserved - to be defined in a future version of
MATLAB

```

```

% handles    structure with handles and user data (see GUIDATA)
import = importdata('-pastespecial');

data = get(handles.Results_Table, 'Data');
if length(import) == (length(data(:,2)) - 1)
    data(2:length(data(:,2)),2) = num2cell(import);

    set(handles.Results_Table, 'Data', data);
else
    %Display a status window saying there was a problem
    set(handles.statusBox, 'String', 'Incompatible array');
end

% --- Executes on button press in glycanSearchButton.
function glycanSearchButton_Callback(hObject, eventdata,
handles)
% hObject    handle to glycanSearchButton (see GCBO)
% eventdata  reserved - to be defined in a future version of
MATLAB
% handles    structure with handles and user data (see GUIDATA)

prompt = {'Enter Glycan:'};
dig_title = 'Glycan Search';
num_lines = 1;
defaultans = {};
entry = inputdlg(prompt,dig_title);

answer = strcat(entry, '[g]');
[junk1, junk2, glycans] =
xlsread('GlycoSheet_Global_CRN.xlsx', 'Metabolite
List', 'A1:K278');

noresult = 0;

for i = 1:length(glycans)
    if strcmp(answer, glycans{i,1}) == 1
        glycan = glycans{i,1};
        description = glycans{i,2};
        formula = glycans{i,3};
        keggID = glycans{i,7};
        pubChemID = glycans{i,8};
        inchi = glycans{i,10};
        smiles = glycans{i,11};
    end
end

```

```

        formatString = 'Glycan:\n%s \n\nDescription\n%s
\n\nFormula:\n%s \n\nKEGG ID:\n%s \n\nPubChem ID:\n%d \n\nInchi
String:\n%s \n\nSMILES:\n%s';
        output = sprintf(formatString, glycan, description,
formula, keggID, pubChemID, inchi, smiles);

        h = msgbox(output)

        break;
    else
        if i == length(glycans)
            noresult = 1;
        end
    end
end
end

if noresult == 1
    h = msgbox('Invalid Glycan: Please try again');
end
end

```

C.2.2 Enzyme Analysis Mode: Exploration Mode (File – MAIN_CRN_ExplorationMode.m)

```

%Runner file for central reaction network model given a
set of enzyme capacity parameters.

function y = MAIN_CRN_ExplorationMode(sheetInput)
h = waitbar(0.25, 'Please Wait');

file = 'GlycoSheet_Global_CRN.xlsx';

sheetInput(1,:) = [];

xlswrite(file, sheetInput, 'Sheet Setup', 'A2');

[parameters] = cell2mat(sheetInput(:,2));

Enzymes = sheetInput(:,1);

```

```

[trash, Reactions] = xlsread(file, 'Reaction
List', 'C2:C542');

cbm = xls2model(file);

h = waitbar(0.4);
pause(2);

out = CRN_CreateConstraints(cbm, parameters, Enzymes,
Reactions);
solution = optimizeCbModel(out, 'max');

h = waitbar(0.6);
pause(2);

[minFlux, maxFlux] = fluxVariability(out);
[Min, Max, All] = CRN_PrintResults(solution, minFlux,
maxFlux, Reactions);

h = waitbar(0.8);
pause(2);

cleararray = cell(1000,3);
xlswrite(file, cleararray, 'Results', 'A2');
xlswrite(file, All, 'Results', 'A2');

disp('Complete');

h = waitbar(1);

y = All;
close(h);
end

```

C.2.3 Enzyme Analysis Mode: Mapping Mode (Algorithm #1) (File – MAIN_CRN_MappingModeAlg1.m)

%Takes a pre-loaded glycoform and fits parameters to the dataset to ensure the best fit using Algorithm #1.

```

function y = MAIN_CRN_MappingModeAlg1(sheetInput)

h = waitbar(0.25, 'Please Wait');

file = 'GlycoSheet_Global_CRN.xlsx';

sheetInput(1,:) = [];

xlswrite(file, sheetInput, 'Sheet Setup', 'D2');

h = waitbar(0.4);
pause(2);

cbm = xls2model(file);

[solution] = optimizeCbModel(cbm, 'max');

h = waitbar(0.6);
pause(2);

roughParams = CRN_CalibrateParams(file, cbm, solution);

h = waitbar(0.8);
pause(2);

[EnzN EnzS params] = xlsread(file, 'Sheet
Setup', 'A1:B18');
params(2:length(params), 2) = num2cell(roughParams);

y = params;

xlswrite(file, y, 'Sheet Setup', 'A1');

h = waitbar(1);

close(h);
end

```

C.2.4 Enzyme Analysis Mode: Mapping Mode (Algorithm #2) (File – MAIN_CRN_MappingModeAlg2.m)

%Takes a pre-loaded glycoform and fits parameters to the dataset to ensure the best fit using Algorithm #2

```
function y = MAIN_CRN_MappingModeAlg2( sheetInput )  
%Applies the second Algorithm to the CRN dataset
```

```
h = waitbar(0.1, 'Please Wait');
```

```
file = 'GlycoSheet_Global_CRN.xlsx';
```

```
sheetInput(1,:) = [];
```

```
xlswrite(file, sheetInput, 'Sheet Setup', 'D2');
```

```
h = waitbar(0.2);
```

```
pause(2);
```

```
%cbm = xls2model(file);
```

```
[inN, inS, Glycans] = xlsread(file, 'Sheet  
Setup', 'D2:E21');
```

```
[inN, inS, Rxns] = xlsread(file, 'Reaction  
List', 'C2:C542');
```

```
[EnzN, EnzS params] = xlsread(file, 'Sheet  
Setup', 'A1:B18');
```

```
EnzS(1,:) = [];
```

```
[P, E, param0, E0, errorTrend, cbm, FBA] =  
CRN_MetricOptimizer(file, Glycans, Rxns, EnzS, h);
```

```
h = waitbar(0.8);
```

```
pause(2);
```

```
roughParams = P;
```

```
params(2:length(params),2) = num2cell(roughParams);
```

```
y = params;
```

```

xlswrite(file, y, 'Sheet Setup', 'A1');

h = waitbar(1);

close(h);

end

```

C.2.5 Enzyme Analysis Mode: Parameter Calibration (File – CRN_CalibrateParams.m)

```

%Takes in the input glycans from the excel sheet and
creates the parameters needed to reproduce the necessary
result.

```

```

%This assumes you already loaded the glycoform into the
correct model space

```

```

function [ y ] = CCM_CalibrateParams( FBA )

```

```

params = zeros(13,1);

```

```

params(1) = FBA.x(1);

```

```

params(2) = FBA.x(2);

```

```

params(3) = FBA.x(3);

```

```

params(4) = FBA.x(4);

```

```

params(5) = FBA.x(5);

```

```

params(6) = FBA.x(6);

```

```

%%Positive value = leaving main branch

```

```

if FBA.x(25) ~= 0
    params(7) = FBA.x(25);
else
    params(7) = FBA.x(24) * -1;
end

%params(8) = Glucose to Mannose transfer branch
%%Positive value = leaving main branch

if FBA.x(13) ~= 0
    params(8) = FBA.x(13);
else
    params(8) = FBA.x(12) * -1;
end

%params(9) = Mannose to fucose transfer branch
%%Positive value = leaving main branch
if FBA.x(9) ~= 0
    params(9) = FBA.x(9);
else
    params(9) = FBA.x(8) * -1;
end

%params(10) = glucose main branch
params(10) = FBA.x(29);

%params(11) = UDP-GlcNAc to CMP-NeuAc transfer
params(11) = FBA.x(34);

%params(12) = ManNAc Recycle
params(12) = FBA.x(35);

%params(13) = GDP-Man to GDP-Fuc
params(13) = FBA.x(17);

y = params;

end

```

C.2.6 Enzyme Analysis Mode: Upper and Lower Bound Generator (File – CRN_CreateConstraints.m)

```
%Takes in a list of ECP values and generates the correct upper and lower bounds for the model (pre-loaded);
```

```
function [ y ] = CRN_CreateConstraints( model, params, EnzS, Reaction_List )
```

```
%Go through and set upper bound of each reaction to the level of the corresponding enzyme.
```

```
%
```

```
%Make Array For enzyme names and parameter values
```

```
for i = 2:length(model.rxns)
    %For each reaction, get the upper bound from the appropriate parameter value and update it with the COBRA function.
    rxnEnz = {model.rxns{i}(1:strfind(model.rxns{i},'-')-1)};
    isSinkRxn = {model.rxns{i}(1:strfind(model.rxns{i},'_')-1)};

    %%
    %Case 1: Upper Bound
    if strcmp(isSinkRxn,'SINK') == 0
        for j = 1:length(params)

            if strcmp(EnzS{j},rxnEnz) == 1
                model = changeRxnBounds(model, model.rxns(i), 10*params(j)*model.ub(1)/100, 'u');
            end
        end
    else
        %If it is Sink, unbound it (both high and low)
        model = changeRxnBounds(model, model.rxns(i), model.ub(1), 'u');
    end
end
```

```

        model = changeRxnBounds(model, model.rxns(i), 0,
'1');

    end

    %%
    %Case 2: Lower Bounds
    %First, extract the glycan you are looking for
    if strcmp(isSinkRxn, 'SINK') == 0

        chop = strfind(Reaction_List{i}, ' ->');
        glycan = {Reaction_List{i}(1:chop-1)};

        prodglycan = strcat({'-> '}, glycan);
        reacglycan = strcat(glycan, {' ->'});

        %Next, find all instances of that glycan on both
product and reactant side

        ProdIndexC = strfind(Reaction_List(1:i-
1), prodglycan);
        ProdIndexes = find(not(cellfun('isempty',
ProdIndexC)));

        if ~isempty(ProdIndexes)
            producedFlux = 0;
            for j = 1:length(ProdIndexes)
                producedFlux = producedFlux +
model.lb(ProdIndexes(j));
            end
        else
            producedFlux = 0;
        end

        ReacIndexC = strfind(Reaction_List(1:i-
1), reacglycan);
        ReacIndexes = find(not(cellfun('isempty',
ReacIndexC)));

        if ~isempty(ReacIndexes)
            consumedFlux = 0;
            for j = 1:length(ReacIndexes)
                consumedFlux = consumedFlux +
model.lb(ReacIndexes(j));
        end
    end

```

```

        end
    else
        consumedFlux = 0;
    end

    if (model.ub(i) - 0.1*model.ub(i)) < 0
        compare = 0;
    else
        compare = (model.ub(i) - 0.1*model.ub(i));
    end

    if (producedFlux - consumedFlux) > (compare)
        model.lb(i) = model.ub(i) - 0.1*model.ub(i);

        if model.lb(i) < 0
            model.lb(i) = 0;
        end
    else
        if producedFlux - consumedFlux >= 0
            model.lb(i) = producedFlux -
consumedFlux;
        else
            model.lb(i) = (0);
        end
    end
end

end

% Remove the Objective Function

rxnNameList = model.rxns{1};

model = changeObjective(model, rxnNameList);

y = model;
end

```

C.2.7 Enzyme Analysis Mode: Outer Loop Management for Mapping Mode Alg2 (File – CRN_MetricOptimizer.m)

```
function [ P , E, param0, E0, errorTrend, cbm, FBA] =  
CRN_MetricOptimizer(file, Glycans, Rxns, EnzS, h)  
%  
% Runs outer algorithm for error minimization during  
mapping mode  
  
%Makes unconstrained Model  
cbm = xls2model(file);  
  
disp('Model Loaded');  
  
% Get optimal flux solution with no enzyme restrictions  
  
[FBA] = optimizeCbModel(cbm, 'min');  
  
%Finds parameters that are 0  
params = CRN_CalibrateParams(file, cbm, FBA);  
  
nonzeros = find(params); %finds all nonzero indeces  
%The nonzeros vector will be what the random number  
generator picks to change  
  
%Gets the initial error from this guess  
  
out = CRN_CreateConstraints(cbm, params, EnzS, Rxns);  
  
[minFlux, maxFlux] = fluxVariability(out,95);  
[Min, Max] = CRN_PrintResults(FBA, minFlux, maxFlux,  
Rxns);  
  
%Initial error (For comparison to all randoms)  
E0 = CRN_FindAdvancedError(Max, Min, Glycans);  
param0 = params;  
  
errorTrend = zeros(1700,1);  
index = 1;
```

```

%Will store 100 final errors and corressponding
parameters in arrays
E = zeros(100,1);
P = zeros(length(params),100);

%% Optimization Loop

%This will go and, starting with all parameters at 0,
increment all eligible parameters (stored in the vector
'nonzeros' by 1 and find the minimum error. Then it will
move onto the next parameter

params = zeros(17,1);
P = params;

h = waitbar(0.3);

for i = 1:length(nonzeros)

    minReached = 0;

    %Makes a new model and finds the error
    out = CRN_CreateConstraints(cbm, params, EnzS, Rxns);
    [minFlux, maxFlux] = fluxVariability(out,95);
    [Min, Max, All] = CRN_PrintResults(FBA, minFlux,
maxFlux, Rxns);
    E = CRN_FindAdvancedError2(Max, Min, Glycans,
nonzeros(i));

    %indexing to make a plot
    errorTrend(index) = CRN_FindAdvancedError(Max, Min,
Glycans);
    index = index + 1;

    while minReached == 0

        if params(nonzeros(i)) < 10
            params(nonzeros(i)) = params(nonzeros(i)) +
1;

            out = CRN_CreateConstraints(cbm, params,
EnzS, Rxns);
            [minFlux, maxFlux] = fluxVariability(out,95);

```

```

        [Min, Max] = CRN_PrintResults(FBA, minFlux,
maxFlux, Rxns);

        Etest = CRN_FindAdvancedError2(Max, Min,
Glycans, nonzeros(i));

        if Etest < E
            E = Etest;
            P = params;

            %indexing to make a plot
            errorTrend(index) =
CRN_FindAdvancedError(Max, Min, Glycans);
            index = index + 1;
        else
            params(nonzeros(i)) = params(nonzeros(i))
- 1;
            minReached = 1;
        end
    else
        minReached = 1;
        disp('Maxed out');
    end

end

h = waitbar(0.3 + 0.5*(i/length(nonzeros)));
end

end

```

C.2.8 Enzyme Analysis Mode: Indexing Error Calculator (File – CRN_FindAdvancedError.m)

```
%Inputs: max, min: outputs from the fluxVariability
command [Name Value]
%Glycan_List: Target Glycoform

%OUTPUT: Error metric for indexing purposes (Not relevant
for model
%function)

function [ error ] = CRN_FindAdvancedError( max, min,
Glycan_List )

%Takes in the maximum and minimum values of all secreted
fluxes and
%evaluates the custom error metrics, reports it

%The Following Metrics Are Calculated (Enzyme Param in
Parentheses):

%1) (ManI) Metric1 = [M5-] / [M6+]
%2) (GnT1) Metric2 = [* (B1)*] / [non-(B1)]
%3) (ManII_B1) Metric3 = [M4-] / [M5]
%4) (ManII_B2) Metric4 = [M3] / [M4]
%5) (GnTII) Metric5 = [* {B*}*] / [non-{B*}]
%6) (GnTIII) Metric6 = [BM*] / [non-BM*]
%7) (GnTIV) Metric7 = [* (B2)*] / [* (B1-)*]
%8) (GnTV) Metric8 = [* {B2}*] / [* {B1-}*]
%9) (FucT) Metric9 = [FM*] / [non-FM*]
%10) (GalT_B1) Metric10 = [*G1+] / [non-galactosylated]
%11) (GalT_B2) Metric11 = [*G2+] / [G1,G0]
%12) (GalT_B3) Metric12 = [*G3+] / [G2,G1,G0]
%13) (GalT_B4) Metric13 = [*G4] / [G3,G2,G1,G0]
%14) (SialT_B1) Metric10 = [*S1+] / [non-sialylated]
%15) (SialT_B2) Metric11 = [*S2+] / [S1,S0]
%16) (SialT_B3) Metric12 = [*S3+] / [S2,S1,S0]
%17) (SialT_B4) Metric13 = [*S4] / [S3,S2,S1,S0]

%----Pseudocode----
%Flow for each For Loop
```

```

%Step 1: Extract all secreted glycans
%Determine if secreted or intracellular
%If secreteed, store in separate matrix
%Find max and min for that glycan (if no min, assume 0)
%Get median value and store in separate array.
%End result is all secreted glycans with median values

%Step 2: Get the metric for target glycoform and
experiment
%Search all glycans, catalogue each one with correct
attribute
%Store in result array, Col1 for target, Col2 for
experimental

%Step 3: Find Error metric for FBA solution
%Get Sum of squared error for all metrics, output as
result.
%-----

expPoolStr = cell(length(max(:,1)),1);
expPoolNum = zeros(length(max(:,1)),1);

%% STEP 0: Remove all null values from glycan List (pre-
processing)

dummy = Glycan_List(:,1);
count = 1;
while count <= length(Glycan_List(:,1))

    if isequal(Glycan_List(count,1),{0}) == 1
        Glycan_List(count,:) = [];
        count = count - 1;
    end

    count = count + 1;

end

%% STEP 1: Extract all secreted glycans

count = 1;
for i = 1:length(max(:,1))

```

```

    if length(max{i,1}) - strfind(max{i,1}, '>') == 0
        %If it makes it in here, we know the glycan is
secreted

        %Now checks to see if the minimum value of a
glycan is zero. If so, leaves it out of the storage

        if
isempty(cell2mat(min(find(strcmp(max{i,1},min(:,1))),2)))
== 0
            maxValue = cell2mat(max(i,2));
            minValue =
cell2mat(min(find(strcmp(max{i,1},min(:,1))),2));

            %Stores all the values
            expPoolStr(count) = max(i,1);
            expPoolNum(count) = minValue + (maxValue -
minValue) / 2;
            count = count + 1;
        end

    end
end

%Now trim all the excess rows of expPoolStr and
expPoolNum

expPoolStr = expPoolStr(1:count - 1);
expPoolNum = expPoolNum(1:count - 1);

%% STEP 2: EXPERIMENTAL ERROR

%This matrix will be used to store all needed information
for each metric efficiently. Each row is a different
metric, each column omposes the glycans in the numerator
(Col 1) and the denominator (Col 2) that make up the
ratio for that metric. Depending on the properties of the
current glycan it will increment appropriately.

binaryTarget = zeros(17,2);
count = 0; %Counter for moving through the array

```

```

%First for target glycoform
for i = 1:length(Glycan_List(:,1))

    %1) (ManI) Metric1 = [M5-] / [M6+]

    count = 1;

    if isempty(strfind(Glycan_List{i,1}, 'M5')) && ...
        isempty(strfind(Glycan_List{i,1}, 'M4')) &&
    ...
        isempty(strfind(Glycan_List{i,1}, 'M3'))
        %This means the glycan is M6, M7, M8, M9
        binaryTarget(count,2) = binaryTarget(count,2) +
cell2mat(Glycan_List(i,2));
    else
        %Else it is M3, M4, M5
        binaryTarget(count,1) = binaryTarget(count,1) +
cell2mat(Glycan_List(i,2));
    end

    %2) (GnT1) Metric2 = [* (B1) *] / [M5]]

    count = 2;

    if ~isempty(strfind(Glycan_List{i,1}, 'M5')) && ...
        isempty(strfind(Glycan_List{i,1}, '(B)'))

        %This means the target is M5
        binaryTarget(count,2) = binaryTarget(count,2) +
cell2mat(Glycan_List(i,2));
    else
        if ~isempty(strfind(Glycan_List{i,1}, '(B)'))
            %Else it has been acted on by GnT1
            binaryTarget(count,1) = binaryTarget(count,1) +
cell2mat(Glycan_List(i,2));
        end
    end

    %3) (ManII_B1) Metric3 = [M4-] / [M5]

    count = 3;

```

```

    if ~isempty(strfind(Glycan_List{i,1},'M5'))
        %This means it is an M5 glycan
        binaryTarget(count,2) = binaryTarget(count,2) +
cell2mat(Glycan_List(i,2));
    else
        if ~isempty(strfind(Glycan_List{i,1},'M4')) ||
...
            ~isempty(strfind(Glycan_List{i,1},'M3'))
            %This means it is M4 or M3
            binaryTarget(count,1) = binaryTarget(count,1) +
+ cell2mat(Glycan_List(i,2));
        end

    end

    %4) (ManII_B2) Metric4 = [M3] / [M4]

    count = 4;

    if ~isempty(strfind(Glycan_List{i,1},'M4'))
        %This means it is M4
        binaryTarget(count,2) = binaryTarget(count,2) +
cell2mat(Glycan_List(i,2));
    else
        if ~isempty(strfind(Glycan_List{i,1},'M3'))
            %It's M3
            binaryTarget(count,1) = binaryTarget(count,1) +
cell2mat(Glycan_List(i,2));
        end
    end

    %5) (GnTII) Metric5 = [*{B*}] / [(B*)]

    count = 5;

    if isempty(strfind(Glycan_List{i,1},{B1})) && ...
        isempty(strfind(Glycan_List{i,1},{B2})) &&
...
        ~isempty(strfind(Glycan_List{i,1},'B'))
        %This means the target has a GnTI GlcNAc but not
yet a GnTII GlcNAc

```

```

        binaryTarget(count,2) = binaryTarget(count,2) +
cell2mat(Glycan_List(i,2));
    else
        if ~isempty(strfind(Glycan_List{i,1},'{B1}')) ||
...
        ~isempty(strfind(Glycan_List{i,1},'{B2}'))

            %Else it has been acted on by GnTII
            binaryTarget(count,1) = binaryTarget(count,1)
+ cell2mat(Glycan_List(i,2));
        end
    end

    %6) (GnTIII) Metric6 = [BM*] / [non-BM*]

    count = 6;

    if isempty(strfind(Glycan_List{i,1},'BM')) && ...
        isempty(strfind(Glycan_List{i,1},'BFM'))
        %This means it is not bisecting
        binaryTarget(count,2) = binaryTarget(count,2) +
cell2mat(Glycan_List(i,2));
    else
        %This means it is bisecting
        binaryTarget(count,1) = binaryTarget(count,1) +
cell2mat(Glycan_List(i,2));
    end

    %7) (GnTIV) Metric7 = [* (B2)*] / [* (B1)*]

    count = 7;

    if ~isempty(strfind(Glycan_List{i,1},'(B1)'))
        %This means it is not acted on by GnTIV yet
        binaryTarget(count,2) = binaryTarget(count,2) +
cell2mat(Glycan_List(i,2));
    else
        if ~isempty(strfind(Glycan_List{i,1},'(B2)'))
            %That mean it is acted on my GnTIV
            binaryTarget(count,1) = binaryTarget(count,1)
+ cell2mat(Glycan_List(i,2));
        end
    end
end

```

```

%8) (GnTV) Metric8 = [*{B2}*] / [*{B1-}*]

count = 8;

if ~isempty(strfind(Glycan_List{i,1}, '{B1}'))
    %This means it is not acted on by GnTV yet
    binaryTarget(count,2) = binaryTarget(count,2) +
cell2mat(Glycan_List(i,2));
else
    if ~isempty(strfind(Glycan_List{i,1}, '{B2}'))
        %That means it is acted on by GnTV
        binaryTarget(count,1) = binaryTarget(count,1)
+ cell2mat(Glycan_List(i,2));
    end
end

%9) (FucT) Metric9 = [FM*] / [non-FM*]

count = 9;

if isempty(strfind(Glycan_List{i,1}, 'F')) && ...
    ~isempty(strfind(Glycan_List{i,1}, '(B)'))

    %This means it is not acted on by FucT yet but
has GnTI GlcNAc
    binaryTarget(count,2) = binaryTarget(count,2) +
cell2mat(Glycan_List(i,2));
else
    if ~isempty(strfind(Glycan_List{i,1}, 'F'))
        %This means it is fucosylated
        binaryTarget(count,1) = binaryTarget(count,1)
+ cell2mat(Glycan_List(i,2));
    end
end

%10) (GalT_B1) Metric10 = [*G1+] / [non-
galactosylated]

count = 10;

if isempty(strfind(Glycan_List{i,1}, 'G')) && ...
    ~isempty(strfind(Glycan_List{i,1}, '(B)'))

```

```

        %This means it is not acted on by GalT yet but
has at least one GlcNAc
        binaryTarget(count,2) = binaryTarget(count,2) +
cell2mat(Glycan_List(i,2));
    else
        if ~isempty(strfind(Glycan_List{i,1},'G'))
            %This means it is galactosylated
            binaryTarget(count,1) = binaryTarget(count,1)
+ cell2mat(Glycan_List(i,2));
        end
    end

%11) (GalT_B2) Metric11 = [*G2+] / [G1]

count = 11;

    if ~isempty(strfind(Glycan_List{i,1},'G1'))
        %This means it only has 1 branch
        binaryTarget(count,2) = binaryTarget(count,2) +
cell2mat(Glycan_List(i,2));
    else
        if ~isempty(strfind(Glycan_List{i,1},'G'))
            %That means it has to have more than 1 branch
by logic
            binaryTarget(count,1) = binaryTarget(count,1)
+ cell2mat(Glycan_List(i,2));
        end
    end

%12) (GalT_B3) Metric12 = [*G3+] / [G2]

count = 12;

    if ~isempty(strfind(Glycan_List{i,1},'G2'))
        %This means it only has 2 branches
        binaryTarget(count,2) = binaryTarget(count,2) +
cell2mat(Glycan_List(i,2));
    else
        if ~isempty(strfind(Glycan_List{i,1},'G3')) ||
...
            ~isempty(strfind(Glycan_List{i,1},'G4'))
            %That means it has 3 or 4 branches

```

```

        binaryTarget(count,1) = binaryTarget(count,1)
+ cell2mat(Glycan_List(i,2));
        end
    end

    %13) (GalT_B4) Metric13 = [*G4] / [G3]

    count = 13;

    if ~isempty(strfind(Glycan_List{i,1},'G3'))
        %This means it only has 3 branches
        binaryTarget(count,2) = binaryTarget(count,2) +
cell2mat(Glycan_List(i,2));
    else
        if ~isempty(strfind(Glycan_List{i,1},'G4'))
            %That means it has 4 branches
            binaryTarget(count,1) = binaryTarget(count,1)
+ cell2mat(Glycan_List(i,2));
        end
    end

    %14) (SialT_B1) Metric10 = [*S1+] / [non-sialylated]

    count = 14;

    if isempty(strfind(Glycan_List{i,1},'S')) && ...
        ~isempty(strfind(Glycan_List{i,1},'G'))
        %This means it is not acted on by SialT yet but
is galactosylated
        binaryTarget(count,2) = binaryTarget(count,2) +
cell2mat(Glycan_List(i,2));
    else
        if ~isempty(strfind(Glycan_List{i,1},'S'))
            %This means it is sialylated
            binaryTarget(count,1) = binaryTarget(count,1)
+ cell2mat(Glycan_List(i,2));
        end
    end

    %15) (SialT_B2) Metric11 = [*S2+] / [S1,S0]

    count = 15;

```

```

    if ~isempty(strfind(Glycan_List{i,1},'S1'))
        %This means it only has 1 branch
        binaryTarget(count,2) = binaryTarget(count,2) +
cell2mat(Glycan_List(i,2));
    else
        if ~isempty(strfind(Glycan_List{i,1},'S'))
            %That means it has to have more than 1 branch
by logic
            binaryTarget(count,1) = binaryTarget(count,1)
+ cell2mat(Glycan_List(i,2));
        end
    end

    %16) (SialT_B3) Metric12 = [*S3+] / [S2,S1,S0]

count = 16;

    if ~isempty(strfind(Glycan_List{i,1},'S2'))
        %This means it only has 2 branches
        binaryTarget(count,2) = binaryTarget(count,2) +
cell2mat(Glycan_List(i,2));
    else
        if ~isempty(strfind(Glycan_List{i,1},'S3')) ||
...
            ~isempty(strfind(Glycan_List{i,1},'S4'))
            %That means it has 3 or 4 branches
            binaryTarget(count,1) = binaryTarget(count,1)
+ cell2mat(Glycan_List(i,2));
        end
    end

    %17) (SialT_B4) Metric13 = [*S4] / [S3,S2,S1,S0]

count = 17;

    if ~isempty(strfind(Glycan_List{i,1},'S3'))
        %This means it only has 3 branches
        binaryTarget(count,2) = binaryTarget(count,2) +
cell2mat(Glycan_List(i,2));
    else
        if ~isempty(strfind(Glycan_List{i,1},'S4'))

```

```

        %That means it has 4 branches
        binaryTarget(count,1) = binaryTarget(count,1)
+ cell2mat(Glycan_List(i,2));
        end
    end

    % disp(binaryTarget);

end

%% MODEL ERROR

%Next for Model output

binaryModel = zeros(17,2);
count = 0;
for i = 1:length(expPoolStr)

    %1) (ManI) Metric1 = [M5-] / [M6+]

    count = 1;

    if isempty(strfind(expPoolStr{i},'M5')) && ...
        isempty(strfind(expPoolStr{i},'M4')) && ...
        isempty(strfind(expPoolStr{i},'M3'))
        %This means the glycan is M6, M7, M8, M9
        binaryModel(count,2) = binaryModel(count,2) +
expPoolNum(i);
    else
        %Else it is M3, M4, M5
        binaryModel(count,1) = binaryModel(count,1) +
expPoolNum(i);
    end

    %2) (GnT1) Metric2 = [* (B1) *] / [M5]]

    count = 2;

    if ~isempty(strfind(expPoolStr{i},'M5')) && ...
        isempty(strfind(expPoolStr{i},' (B)'))
        %This means the target is not ated on by GnT1
        binaryModel(count,2) = binaryModel(count,2) +
expPoolNum(i);

```

```

else
    if ~isempty(strfind(expPoolStr{i}, 'B'))

        %Else it has been acted on by GnT1
        binaryModel(count,1) = binaryModel(count,1) +
expPoolNum(i);
    end
end

%3) (ManII_B1) Metric3 = [M4-] / [M5]

count = 3;

if ~isempty(strfind(expPoolStr{i}, 'M5'))
    %This means it is an M5 glycan
    binaryModel(count,2) = binaryModel(count,2) +
expPoolNum(i);
else
    if ~isempty(strfind(expPoolStr{i}, 'M4')) || ...
        ~isempty(strfind(expPoolStr{i}, 'M3'))
        %This means it is M4 or M3
        binaryModel(count,1) = binaryModel(count,1) +
expPoolNum(i);
    end

end

%4) (ManII_B2) Metric4 = [M3] / [M4]

count = 4;

if ~isempty(strfind(expPoolStr{i}, 'M4'))
    %This means it is M4
    binaryModel(count,2) = binaryModel(count,2) +
expPoolNum(i);
else
    if ~isempty(strfind(expPoolStr{i}, 'M3'))
        %It's M3
        binaryModel(count,1) = binaryModel(count,1) +
expPoolNum(i);
    end
end
end

```

```

%5) (GnTII) Metric5 = [*{B}*] / [non-{B}]

count = 5;

if isempty(strfind(expPoolStr{i}, '{B1}')) && ...
    isempty(strfind(expPoolStr{i}, '{B2}')) && ...
    ~isempty(strfind(expPoolStr{i}, '(B)'))
    %This means the target is not acted on by GnTII
    binaryModel(count,2) = binaryModel(count,2) +
expPoolNum(i);
else
    if ~isempty(strfind(expPoolStr{i}, '{B1}')) || ...
        ~isempty(strfind(expPoolStr{i}, '{B2}'))
        %Else it has been acted on by GnT1
        binaryModel(count,1) = binaryModel(count,1) +
expPoolNum(i);
    end
end

%6) (GnTIII) Metric6 = [BM*] / [non-BM*]

count = 6;

if isempty(strfind(expPoolStr{i}, 'BM')) && ...
    isempty(strfind(expPoolStr{i}, 'BFM'))
    %This means it is not bisecting
    binaryModel(count,2) = binaryModel(count,2) +
expPoolNum(i);
else
    %This means it is bisecting
    binaryModel(count,1) = binaryModel(count,1) +
expPoolNum(i);
end

%7) (GnTIV) Metric7 = [*(B2)*] / [*(B1)*]

count = 7;

if ~isempty(strfind(expPoolStr{i}, '(B1)'))
    %This means it is not acted on by GnTIV yet
    binaryModel(count,2) = binaryModel(count,2) +
expPoolNum(i);

```

```

else
    if ~isempty(strfind(expPoolStr{i}, '(B2)'))
        %That measn it is acted on my GnTIV
        binaryModel(count,1) = binaryModel(count,1) +
expPoolNum(i);
    end
end

%8) (GnTV) Metric8 = [*{B2}*] / [*{B1-*}]

count = 8;

if ~isempty(strfind(expPoolStr{i}, '{B1}'))
    %This means it is not acted on by GnTV yet
    binaryModel(count,2) = binaryModel(count,2) +
expPoolNum(i);
else
    if ~isempty(strfind(expPoolStr{i}, '{B2}'))
        %That measn it is acted on my GnTV
        binaryModel(count,1) = binaryModel(count,1) +
expPoolNum(i);
    end
end

%9) (FucT) Metric9 = [FM*] / [non-FM*]

count = 9;

if isempty(strfind(expPoolStr{i}, 'F')) && ...
    ~isempty(strfind(expPoolStr{i}, '(B)'))
    %This means it is not acted on by FucT yet
    binaryModel(count,2) = binaryModel(count,2) +
expPoolNum(i);
else
    if ~isempty(strfind(expPoolStr{i}, 'F'))
        %This means it is fucosylated
        binaryModel(count,1) = binaryModel(count,1) +
expPoolNum(i);
    end
end
end

```

```

    %10) (GalT_B1) Metric10 = [*G1+] / [non-
galactosylated]

    count = 10;

    if isempty(strfind(expPoolStr{i}, 'G')) && ...
        ~isempty(strfind(expPoolStr{i}, 'B'))
        %This means it is not acted on by GalT yet
        binaryModel(count,2) = binaryModel(count,2) +
expPoolNum(i);
    else
        if ~isempty(strfind(expPoolStr{i}, 'G'))
            %This means it is galactosylated
            binaryModel(count,1) = binaryModel(count,1) +
expPoolNum(i);
        end
    end

    %11) (GalT_B2) Metric11 = [*G2+] / [G1]

    count = 11;

    if ~isempty(strfind(expPoolStr{i}, 'G1'))
        %This means it only has 1 branch
        binaryModel(count,2) = binaryModel(count,2) +
expPoolNum(i);
    else
        if ~isempty(strfind(expPoolStr{i}, 'G'))
            %That means it has to have more than 1 branch
by logic
            binaryModel(count,1) = binaryModel(count,1) +
expPoolNum(i);
        end
    end

    %12) (GalT_B3) Metric12 = [*G3+] / [G2]

    count = 12;

    if ~isempty(strfind(expPoolStr{i}, 'G2'))
        %This means it only has 2 branches
        binaryModel(count,2) = binaryModel(count,2) +
expPoolNum(i);

```

```

else
    if ~isempty(strfind(expPoolStr{i}, 'G3')) || ...
        ~isempty(strfind(expPoolStr{i}, 'G4'))
        %That means it has 3 or 4 branches
        binaryModel(count,1) = binaryModel(count,1) +
expPoolNum(i);
    end
end

%13) (GalT_B4) Metric13 = [*G4] / [G3]

count = 13;

if ~isempty(strfind(expPoolStr{i}, 'G3'))
    %This means it only has 3 branches
    binaryModel(count,2) = binaryModel(count,2) +
expPoolNum(i);
else
    if ~isempty(strfind(expPoolStr{i}, 'G4'))
        %That means it has 4 branches
        binaryModel(count,1) = binaryModel(count,1) +
expPoolNum(i);
    end
end

%14) (SialT_B1) Metric10 = [*S1+] / [non-sialylated]

count = 14;

if isempty(strfind(expPoolStr{i}, 'S')) && ...
    ~isempty(strfind(expPoolStr{i}, 'G'))
    %This means it is not acted on by SialT yet but
is galactosylated
    binaryModel(count,2) = binaryModel(count,2) +
expPoolNum(i);
else
    if ~isempty(strfind(expPoolStr{i}, 'S'))
        %This means it is sialylated
        binaryModel(count,1) = binaryModel(count,1) +
expPoolNum(i);
    end
end
end

```

```

%15) (SialT_B2) Metric11 = [*S2+] / [S1,S0]

count = 15;

if ~isempty(strfind(expPoolStr{i},'S1'))
    %This means it only has 1 branch
    binaryModel(count,2) = binaryModel(count,2) +
expPoolNum(i);
else
    if ~isempty(strfind(expPoolStr{i},'S'))
        %That means it has to have more than 1 branch
by logic
        binaryModel(count,1) = binaryModel(count,1) +
expPoolNum(i);
    end
end

%16) (SialT_B3) Metric12 = [*S3+] / [S2,S1,S0]

count = 16;

if ~isempty(strfind(expPoolStr{i},'S2'))
    %This means it only has 2 branches
    binaryModel(count,2) = binaryModel(count,2) +
expPoolNum(i);
else
    if ~isempty(strfind(expPoolStr{i},'S3')) || ...
        ~isempty(strfind(expPoolStr{i},'S4'))
        %That means it has 3 or 4 branches
        binaryModel(count,1) = binaryModel(count,1) +
expPoolNum(i);
    end
end

%17) (SialT_B4) Metric13 = [*S4] / [S3,S2,S1,S0]

count = 17;

if ~isempty(strfind(expPoolStr{i},'S3'))
    %This means it only has 3 branches

```

```

        binaryModel(count,2) = binaryModel(count,2) +
expPoolNum(i);
    else
        if ~isempty(strfind(expPoolStr{i},'S4'))
            %That means it has 4 branches
            binaryModel(count,1) = binaryModel(count,1) +
expPoolNum(i);
        end
    end

    % disp(binaryModel);
end

```

```

%% Calculate Final Metric Values

```

```

%Divide first column of binaryMatrices by the second
column to get the ratio.

```

```

targetResult = zeros(17,1);
expResult = zeros(17,1);

```

```

for i = 1:length(binaryTarget(:,1))

    if binaryTarget(i,1) == 0 && binaryTarget(i,2) == 0

        targetResult(i) = 0;
    else
        targetResult(i) = (binaryTarget(i,1) /
(binaryTarget(i,1) + binaryTarget(i,2))) * 100;
    end

end

```

```

for i = 1:length(binaryModel(:,1))

    if binaryModel(i,1) == 0 && binaryModel(i,2) == 0

        expResult(i) = 0;
    else
        expResult(i) = (binaryModel(i,1) /
(binaryModel(i,1) + binaryModel(i,2))) * 100;
    end
end

```

```

    end
end

%Place where all metrics are stored

error = 0;
for i = 1:17
    error = error + (targetResult(i) - expResult(i))^2 ;
end

end

```

C.2.9 Enzyme Analysis Mode: Model Error Metric Calculator (File – CRN_FindAdvancedError2.m)

```

%Inputs: max, min: outputs from the fluxVariability
command [Name Value]
%Glycan_List: Target Glycoform

%OUTPUT: Error metric for calculation purposes (used in
model)

function [ error ] = CRN_FindAdvancedError2( max, min,
Glycan_List, currentEnzyme )

%Takes in the maximum and minimum values of all secreted
fluxes and
%evaluates the custom error metrics, reports it

%The Following Metrics Are Calculated (Enzyme Param in
Parentheses):

%1) (ManI) Metric1 = [M5-] / [M6+]
%2) (GnT1) Metric2 = [* (B1)*] / [non- (B1)]
%3) (ManII_B1) Metric3 = [M4-] / [M5]
%4) (ManII_B2) Metric4 = [M3] / [M4]

```

```

%5) (GnTIII) Metric5 = [*{B*}*] / [non-{B*}]
%6) (GnTIII) Metric6 = [BM*] / [non-BM*]
%7) (GnTIV) Metric7 = [*{B2}*] / [*{B1-}*]
%8) (GnTV) Metric8 = [*{B2}*] / [*{B1-}*]
%9) (FucT) Metric9 = [FM*] / [non-FM*]
%10) (GalT_B1) Metric10 = [*G1+] / [non-galactosylated]
%11) (GalT_B2) Metric11 = [*G2+] / [G1,G0]
%12) (GalT_B3) Metric12 = [*G3+] / [G2,G1,G0]
%13) (GalT_B4) Metric13 = [*G4] / [G3,G2,G1,G0]
%14) (SialT_B1) Metric10 = [*S1+] / [non-sialylated]
%15) (SialT_B2) Metric11 = [*S2+] / [S1,S0]
%16) (SialT_B3) Metric12 = [*S3+] / [S2,S1,S0]
%17) (SialT_B4) Metric13 = [*S4] / [S3,S2,S1,S0]

%----Pseudocode----
%Flow for each For Loop

%Step 1: Extract all secreted glycans
%Determine if secreted or intracellular
%If secreted, store in separate matrix
%Find max and min for that glycan (if no min, assume 0)
%Get median value and store in separate array.
%End result is all secreted glycans with median values

%Step 2: Get the metric for target glycoform and
experiment
%Search all glycans, catalogue each one with correct
attribute
%Store in result array, Col 1 for target, Col2 for
experimental

%Step 3: Find error metric for FBA solution
%Get sum of squared error for all metrics, output as
result.
%-----

expPoolStr = cell(length(max(:,1)),1);
expPoolNum = zeros(length(max(:,1)),1);

%% STEP 0: Remove all null values from glycan List (pre-
processing)

dummy = Glycan_List(:,1);

```

```

count = 1;
while count <= length(Glycan_List(:,1))

    if isequal(Glycan_List(count,1),{0}) == 1
        Glycan_List(count,:) = [];
        count = count - 1;
    end

    count = count + 1;

end

%% STEP 1: Extract all secreted glycans

count = 1;
for i = 1:length(max(:,1))
    if length(max{i,1}) - strfind(max{i,1},'>') == 0
        %If it makes it in here, we know the glycan is
secreted

        %Now checks to see if the minimum value of a
glycan is zero. If so,leaves it out of the storage

        if
iseempty(cell2mat(min(find(strcmp(max{i,1},min(:,1))),2)))
== 0
            maxValue = cell2mat(max(i,2));
            minValue =
cell2mat(min(find(strcmp(max{i,1},min(:,1))),2));

            %Stores all the values
            expPoolStr(count) = max(i,1);
            expPoolNum(count) = minValue + (maxValue -
minValue) / 2;
            count = count + 1;

        end

    end
end

%Now trim all the excess rows of expPoolStr and
expPoolNum

```

```

expPoolStr = expPoolStr(1:count - 1);
expPoolNum = expPoolNum(1:count - 1);

%% STEP 2: EXPERIMENTAL ERROR

total = 100;

%This matrix will be used to store all needed information
for each metric efficiently. Each row is a different
metric, each column comprises the glycans in the
numerator (Col 1) and the denominator (Col 2) that make
up the ratio for that metric. Depending on the properties
of the current glycan it will increment appropriately.
binaryTarget = zeros(17,1);
count = 0; %Counter for moving through the array
%First for target glycoform
for i = 1:length(Glycan_List(:,1))

    %1) (ManI) Metric1 = % M5 or less (CHANGED)

    count = 1;

    if currentEnzyme == count
        if ~isempty(strfind(Glycan_List{i,1}, 'M5')) ||
...
            ~isempty(strfind(Glycan_List{i,1}, 'M4'))
|| ...
            ~isempty(strfind(Glycan_List{i,1}, 'M3'))
        %This means the glycan is M5, M4, M3
        binaryTarget(count) = binaryTarget(count) +
cell2mat(Glycan_List(i,2));
    end
end

    %2) (GnT1) Metric2 = % glycans with (B*)

    count = 2;

    if currentEnzyme == count

```

```

        if ~isempty(strfind(Glycan_List{i,1}, 'B'))
            %Else it has been acted on by GnT1
            binaryTarget(count) = binaryTarget(count) +
cell2mat(Glycan_List(i,2));
        end
    end

    %3) (ManII_B1) Metric3 = % M4 or M3

    count = 3;

    if currentEnzyme == count
        if ~isempty(strfind(Glycan_List{i,1}, 'M4')) ||
...
            ~isempty(strfind(Glycan_List{i,1}, 'M3'))
            %This means it is M4 or M3
            binaryTarget(count) = binaryTarget(count) +
cell2mat(Glycan_List(i,2));
        end
    end

    %4) (ManII_B2) Metric4 = % of [M3]

    count = 4;

    if currentEnzyme == count
        if ~isempty(strfind(Glycan_List{i,1}, 'M3'))
            %It's M3
            binaryTarget(count) = binaryTarget(count) +
cell2mat(Glycan_List(i,2));
        end
    end

    %5) (GnTII) Metric5 = % {B*} Glycans

    count = 5;

    if currentEnzyme == count
        if ~isempty(strfind(Glycan_List{i,1}, '{B1}')) ||
...

```

```

        ~isempty(strfind(Glycan_List{i,1}, '{B2}'))

        %it has been acted on by GnTII
        binaryTarget(count) = binaryTarget(count) +
cell2mat(Glycan_List(i,2));
    end
end

%6) (GnTIII) Metric6 = % Bisecting glycans

count = 6;

if currentEnzyme == count
    if ~isempty(strfind(Glycan_List{i,1}, 'BM')) ||
...
        ~isempty(strfind(Glycan_List{i,1}, 'BFM'))
        %This means it is not bisecting
        binaryTarget(count) = binaryTarget(count) +
cell2mat(Glycan_List(i,2));
    end
end

%7) (GnTIV) Metric7 = % (B2) Glycans

count = 7;

if currentEnzyme == count
    if ~isempty(strfind(Glycan_List{i,1}, '(B2)'))
        %That means it is acted on my GnTIV
        binaryTarget(count) = binaryTarget(count) +
cell2mat(Glycan_List(i,2));
    end
end

%8) (GnTV) Metric8 = % {B2} Glycans

count = 8;

if currentEnzyme == count
    if ~isempty(strfind(Glycan_List{i,1}, '{B2}'))
        %That means it is acted on my GnTV
        binaryTarget(count) = binaryTarget(count) +
cell2mat(Glycan_List(i,2));
    end
end

```

```

end

%9) (FucT) Metric9 = % Fucosylated

count = 9;

if currentEnzyme == count
    if ~isempty(strfind(Glycan_List{i,1},'F'))
        %This means it is fucosylated
        binaryTarget(count) = binaryTarget(count) +
cell2mat(Glycan_List(i,2));
    end
end

%10) (GalT_B1) Metric10 = % Galactosylated

count = 10;

if currentEnzyme == count
    if ~isempty(strfind(Glycan_List{i,1},'G'))
        %This means it is galactosylated
        binaryTarget(count) = binaryTarget(count) +
cell2mat(Glycan_List(i,2));
    end
end

%11) (GalT_B2) Metric11 = % double, triple,
quadrouple galactosylated

count = 11;

if currentEnzyme == count
    if ~isempty(strfind(Glycan_List{i,1},'G2')) ||
...
        ~isempty(strfind(Glycan_List{i,1},'G3'))
|| ...
        ~isempty(strfind(Glycan_List{i,1},'G4'))
        %This means it has 2,3,4 branches
        binaryTarget(count) = binaryTarget(count) +
cell2mat(Glycan_List(i,2));
    end
end

```

```

end

%12) (GalT_B3) Metric12 = % triple or quadruple
galactosylated

count = 12;

if currentEnzyme == count
    if ~isempty(strfind(Glycan_List{i,1},'G3')) ||
...
        ~isempty(strfind(Glycan_List{i,1},'G4'))
            %That means it has 3 or 4 branches
            binaryTarget(count) = binaryTarget(count) +
cell2mat(Glycan_List(i,2));
        end
    end
end

%13) (GalT_B4) Metric13 = % quadruple galactosylated

count = 13;

if currentEnzyme == count
    if ~isempty(strfind(Glycan_List{i,1},'G4'))
        %That means it has 4 branches
        binaryTarget(count) = binaryTarget(count) +
cell2mat(Glycan_List(i,2));
    end
end

%14) (SialT_B1) Metric10 = % sialylated

count = 14;

if currentEnzyme == count
    if ~isempty(strfind(Glycan_List{i,1},'S'))
        %This means it is sialylated
        binaryTarget(count) = binaryTarget(count) +
cell2mat(Glycan_List(i,2));
    end
end
end

```

```

    %15) (SialT_B2) Metric11 = % double, triple,
quadrouple galactosylated

    count = 15;

    if currentEnzyme == count
        if ~isempty(strfind(Glycan_List{i,1}, 'S2')) ||
...
            ~isempty(strfind(Glycan_List{i,1}, 'S3'))
|| ...
            ~isempty(strfind(Glycan_List{i,1}, 'S4'))
            %This means it only has 1 branch
            binaryTarget(count) = binaryTarget(count) +
cell2mat(Glycan_List(i,2));
        end
    end

    %16) (SialT_B3) Metric12 = % triple or quadrouple
galactosylated

    count = 16;

    if currentEnzyme == count
        if ~isempty(strfind(Glycan_List{i,1}, 'S3')) ||
...
            ~isempty(strfind(Glycan_List{i,1}, 'S4'))
            %That means it has 3 or 4 branches
            binaryTarget(count) = binaryTarget(count) +
cell2mat(Glycan_List(i,2));
        end
    end

    %17) (SialT_B4) Metric13 = % quadrouply sialylated

    count = 17;

    if currentEnzyme == count
        if ~isempty(strfind(Glycan_List{i,1}, 'S4'))
            %That means it has 4 branches
            binaryTarget(count) = binaryTarget(count) +
cell2mat(Glycan_List(i,2));
        end
    end
end

```

```

        % disp(binaryTarget);

end

%% MODEL ERROR

%Next for Model output

binaryModel = zeros(17,1);
count = 0;
for i = 1:length(expPoolStr)

    %1) (ManI) Metric1 = % M5 or less

    count = 1;

    if currentEnzyme == count
        if ~isempty(strfind(expPoolStr{i}, 'M5')) || ...
            ~isempty(strfind(expPoolStr{i}, 'M4')) ||
...
            ~isempty(strfind(expPoolStr{i}, 'M3'))
            %This means the glycan is M5, M4, M3
            binaryModel(count) = binaryModel(count) +
expPoolNum(i);
        end
    end

    %2) (GnT1) Metric2 = % (B*) glycans

    count = 2;

    if currentEnzyme == count
        if ~isempty(strfind(expPoolStr{i}, 'B'))
            %it has been acted on by GnT1
            binaryModel(count) = binaryModel(count) +
expPoolNum(i);
        end
    end

    %3) (ManII_B1) Metric3 = % of M4 or less

    count = 3;

```

```

    if currentEnzyme == count
        if ~isempty(strfind(expPoolStr{i}, 'M4')) || ...
            ~isempty(strfind(expPoolStr{i}, 'M3'))
            %This means it is M4 or M3
            binaryModel(count) = binaryModel(count) +
expPoolNum(i);
        end
    end

    %4) (ManII_B2) Metric4 = % M3 Glycans

    count = 4;

    if currentEnzyme == count
        if ~isempty(strfind(expPoolStr{i}, 'M3'))
            %It's M3
            binaryModel(count) = binaryModel(count) +
expPoolNum(i);
        end
    end

    %5) (GnTII) Metric5 = % {B*}

    count = 5;

    if currentEnzyme == count
        if ~isempty(strfind(expPoolStr{i}, '{B1}')) || ...
            ~isempty(strfind(expPoolStr{i}, '{B2}'))
            %Else it has been acted on by GnT1
            binaryModel(count) = binaryModel(count) +
expPoolNum(i);
        end
    end

    %6) (GnTIII) Metric6 = % Bisecting Glycans

    count = 6;

    if currentEnzyme == count
        if ~isempty(strfind(expPoolStr{i}, 'BM')) || ...
            ~isempty(strfind(expPoolStr{i}, 'BFM'))

```

```

        %This means it is not bisecting
        binaryModel(count) = binaryModel(count) +
expPoolNum(i);
    end
end

%7) (GnTIV) Metric7 = % (B2) Glycans

count = 7;

if currentEnzyme == count
    if ~isempty(strfind(expPoolStr{i}, '(B2)'))
        %That means it is acted on my GnTIV
        binaryModel(count) = binaryModel(count) +
expPoolNum(i);
    end
end

%8) (GnTV) Metric8 = % {B2} Glycans

count = 8;

if currentEnzyme == count
    if ~isempty(strfind(expPoolStr{i}, '{B2}'))
        %That means it is acted on my GnTV
        binaryModel(count) = binaryModel(count) +
expPoolNum(i);
    end
end

%9) (FucT) Metric9 = % Fucosylated Glycans

count = 9;

if currentEnzyme == count
    if ~isempty(strfind(expPoolStr{i}, 'F'))
        %This means it is fucosylated
        binaryModel(count) = binaryModel(count) +
expPoolNum(i);
    end
end
end

```

```

%10) (GalT_B1) Metric10 = % Galactosylated

count = 10;

if currentEnzyme == count
    if ~isempty(strfind(expPoolStr{i}, 'G'))
        %This means it is galactosylated
        binaryModel(count) = binaryModel(count) +
expPoolNum(i);
    end
end

%11) (GalT_B2) Metric11 = % 2 or more galactosylated
branches

count = 11;

if currentEnzyme == count
    if ~isempty(strfind(expPoolStr{i}, 'G2')) || ...
        ~isempty(strfind(expPoolStr{i}, 'G3')) ||
...
        ~isempty(strfind(expPoolStr{i}, 'G4'))
        %That means it has to have 2 or more branches
        binaryModel(count) = binaryModel(count) +
expPoolNum(i);
    end
end

%12) (GalT_B3) Metric12 = % 3 or more galactosylated

count = 12;

if currentEnzyme == count
    if ~isempty(strfind(expPoolStr{i}, 'G3')) || ...
        ~isempty(strfind(expPoolStr{i}, 'G4'))
        %That means it has 3 or 4 branches
        binaryModel(count) = binaryModel(count) +
expPoolNum(i);
    end
end

```

```

%13) (GalT_B4) Metric13 = % quadrouple-galactosylated

count = 13;

if currentEnzyme == count
    if ~isempty(strfind(expPoolStr{i}, 'G4'))
        %That means it has 4 branches
        binaryModel(count) = binaryModel(count) +
expPoolNum(i);
    end
end

%14) (SialT_B1) Metric10 = % sialylated

count = 14;

if currentEnzyme == count
    if ~isempty(strfind(expPoolStr{i}, 'S'))
        %This means it is sialylated
        binaryModel(count) = binaryModel(count) +
expPoolNum(i);
    end
end

%15) (SialT_B2) Metric11 = % double, triple,
quadrouple sialylated

count = 15;

if currentEnzyme == count
    if ~isempty(strfind(expPoolStr{i}, 'S2')) || ...
        ~isempty(strfind(expPoolStr{i}, 'S3')) ||
...
        ~isempty(strfind(expPoolStr{i}, 'S4'))
        %This means it only has 1 branch
        binaryModel(count) = binaryModel(count) +
expPoolNum(i);
    end
end

```

```

end

%16) (SialT_B3) Metric12 = % triple or quadruple
sialylated

count = 16;

if currentEnzyme == count
    if ~isempty(strfind(expPoolStr{i}, 'S3')) || ...
        ~isempty(strfind(expPoolStr{i}, 'S4'))
        %That means it has 3 or 4 branches
        binaryModel(count) = binaryModel(count) +
expPoolNum(i);
    end
end

%17) (SialT_B4) Metric13 = % quadruple sialylated

count = 17;

if currentEnzyme == count
    if ~isempty(strfind(expPoolStr{i}, 'S4'))
        %That means it has 4 branches
        binaryModel(count) = binaryModel(count) +
expPoolNum(i);
    end
end

% disp(binaryModel);
end

%% Calculate Final Metric Values

%Divide first column of binaryMatrices by the second
column to get the
%ratio.

targetResult = binaryTarget / total;
expResult = binaryModel / total;

error = 0;

```

```

for i = 1:17
    error = error + (targetResult(i) - expResult(i))^2 ;
end

end

```

C.2.10 Enzyme Analysis Mode: Print Results (File – CRN_PrintResults.m)

```

%Prints all the nonzero glycans

function [ minoutput, maxoutput, masteroutput ] =
PrintResults( solution, minF, maxF, rawRxns )

minFlux = minF;
maxFlux = maxF;

count = 1;
names = cell(1);
nums = zeros(1);

output = cell(1);

for i = 1:length(solution.x)
    if solution.x(i) > .001
        output(count,1) = rawRxns(i);
        output(count,2) = num2cell(solution.x(i));
        count = count + 1;
    end
end

count = 1;
for i = 1:length(rawRxns)

    if minFlux(i) > .001
        names(count) = rawRxns(i);
        nums(count) = minFlux(i);
    end
end

```

```

        count = count + 1;
    end
end

minoutput = [names' num2cell(nums)'];

names = cell(1);
nums = zeros(1);

count = 1;
for i = 1:length(rawRxns)

    if maxFlux(i) > .001
        names(count) = rawRxns(i);
        nums(count) = maxFlux(i);
        count = count + 1;
    end
end

maxoutput = [names' num2cell(nums)'];

%Now combines both submatrices into an easy-to-read
matrix and prints it in
%excel.
filled = 0;
mincount = 1;
maxcount = 1;
failsafe = 0;

masteroutput = cell(length(maxoutput(:,1)),3);

%col 1 = names, col 2 = min, col 3 = max
masteroutput(:,1) = maxoutput(:,1);

while filled == 0
    if mincount == length(minoutput(:,1)) + 1
        filled = 1;
    else
        if
strcmp(maxoutput(maxcount,1),minoutput(mincount,1)) == 1
            masteroutput(maxcount,2) =
minoutput(mincount,2);

```

```

        masteroutput(maxcount,3) =
maxoutput(maxcount,2);
        mincount = mincount + 1;
        maxcount = maxcount + 1;
    else
        masteroutput{maxcount,2} = 0;
        masteroutput(maxcount,3) =
maxoutput(maxcount,2);
        maxcount = maxcount + 1;
    end
end
end

failsafe = failsafe + 1;
if failsafe == 10000
    filled = 1;
end
end

end
end

```

C.2.11 Metabolism Analysis Mode: Parameter Calibration (File – MAIN_CCM_ExplorationMode.m)

```

%Runner file for central carbon metabolism model given a
set of
%reaction parameters

function y = MAIN_CCM_ExplorationMode(sheetInput)

h = waitbar(0.25, 'Please Wait');

file = 'GlycoSheet_Global_CCM.xlsm';
[junk1 junk2 Rxns] = xlsread(file, 'Reaction List',
'C2:C50');

if sheetInput{8,2} == 0
    minicalc1 = sheetInput{9,2};
else
    minicalc1 = 0;

```

```

end

cellInsert1 = {'Glc-6-P Split Ratio (adjusted)'
minicalc1};

if sheetInput{12,2} == 0
    minicalc2 = sheetInput{13,2};
else
    minicalc2 = 0;
end

cellInsert2 = {'Fruc-6-P Split Ratio (adjusted)'
minicalc2};

curatedInput = [sheetInput(1:9,:); cellInsert1;
sheetInput(10:12,:); ...
cellInsert2; sheetInput(13:15,:)];

curatedInput(1,:) = [];

xlswrite(file, curatedInput, 'Sheet Setup', 'A2');

[parameters] = cell2mat(curatedInput(:,2));

cbm = xls2model(file);

h = waitbar(0.4);
pause(2);

out = CCM_CreateConstraints(cbm, parameters);

[FBA] = optimizeCbModel(out, 'min');

output = CCM_PrintResults(FBA, Rxns);

h = waitbar(0.8);
pause(2);

```

```

cleararray = cell(1000,2);

xlswrite(file, cleararray, 'Results', 'A2');
xlswrite(file, output, 'Results', 'A2');

h = waitbar(1);

y = output;
close(h);

end

```

C.2.12 Metabolism Analysis Mode: Mapping Mode (File – MAIN_CCM_MappingModeAlg1.m)

```

%MappingModeCCM

%This calibrates parameters to a sample glycoform and
writes them to the
%excel sheet.

function y = MappingModeAlg1(sheetInput, inputObj)

h = waitbar(0.25, 'Please Wait');

file = 'GlycoSheet_Global_CCM.xlsm';

sheetInput(1,:) = [];

xlswrite(file, sheetInput, 'Sheet Setup', 'D2');
xlswrite(file, inputObj, 'Reaction List', 'I3');

h = waitbar(0.4);
pause(2);

cbm = xls2model(file);
[FBA] = optimizeCbModel(cbm, 'min');

```

```

h = waitbar(0.6);
pause(2);

xlswrite(file, FBA.x, 'Results', 'B2');

params = CCM_CalibrateParams(FBA);

h = waitbar(0.8);
pause(2);

xlswrite(file, params, 'Sheet Setup', 'B2');

%

disp('Complete');

[junk1 junk2 data] = xlsread(file, 'Sheet Setup',
'A1:B14');

h = waitbar(1);

close(h);
y = data;

end

```

C.2.13 Metabolism Analysis Mode: Parameter Calibration (File – CCM_CalibrateParams.m)

```

%Takes in the input glycans from the excel sheet and
creates the parameters needed to reproduce the necessary
result

%This assumes you already loaded the glycoform into the
correct model space

%We need to extract 13 parameters from the solution based
on the flux values

```

```

function [ y ] = CCM_CalibrateParams( FBA )

params = zeros(13,1);

%params(1) = GlucoseIN
params(1) = FBA.x(1);

%params(2) = GalactoseIN
params(2) = FBA.x(2);

%params(3) = MannoseIN
params(3) = FBA.x(3);

%params(4) = FucoseIN
params(4) = FBA.x(4);

%params(5) = ManNAcIN
params(5) = FBA.x(5);

%params(6) = GlucosamineIN
params(6) = FBA.x(6);

%params(7) = Glucose to Gal Transfer Branch
%%Positive value = leaving main branch

if FBA.x(25) ~= 0
    params(7) = FBA.x(25);
else
    params(7) = FBA.x(24) * -1;
end

%params(8) = Glucose to Mannose transfer branch
%%Positive value = leaving main branch

if FBA.x(13) ~= 0
    params(8) = FBA.x(13);
else
    params(8) = FBA.x(12) * -1;
end

%params(9) = Mannose to fucose transfera branch
%%Positive value = leaving main branch

```

```

if FBA.x(9) ~= 0
    params(9) = FBA.x(9);
else
    params(9) = FBA.x(8) * -1;
end

%params(10) = glucose main branch
params(10) = FBA.x(29);

%params(11) = UDP-GlcNAc to CMP-NeuAc transfer
params(11) = FBA.x(34);

%params(12) = ManNAc Recycle
params(12) = FBA.x(35);

%params(13) = GDP-Man to GDP-Fuc
params(13) = FBA.x(17);

y = params;

end

```

C.2.14 Metabolism Analysis Mode: Generate Upper and Lower Bounds (File – CCM_CreateConstraints.m)

```

function [y] = CCM_CreateConstraints( model, params )

%This script will create the necessary constraints on the
model based on the input parameters for all reactions for
the CCM
%Also sets the objective, if necessary

%Glucose IN

glucoseIN = params(1);
model = changeRxnBounds(model, model.rxns(1), glucoseIN ,
'u');
model = changeRxnBounds(model, model.rxns(1), glucoseIN,
'l');

```

```

%Galactose IN

galactoseIN = params(2);
model = changeRxnBounds(model, model.rxns(2),
galactoseIN, 'u');
model = changeRxnBounds(model, model.rxns(2),
galactoseIN, 'l');

%Mannose IN

mannoseIN = params(3);
model = changeRxnBounds(model, model.rxns(3), mannoseIN ,
'u');
model = changeRxnBounds(model, model.rxns(3), mannoseIN,
'l');

%Fucose IN

fucoseIN = params(4);
model = changeRxnBounds(model, model.rxns(4), fucoseIN ,
'u');
model = changeRxnBounds(model, model.rxns(4), fucoseIN,
'l');

%MAcNAc IN

manNAcIN = params(5);
model = changeRxnBounds(model, model.rxns(5), manNAcIN ,
'u');
model = changeRxnBounds(model, model.rxns(5), manNAcIN,
'l');

%Glucosamine IN

glucosamineIN = params(6);
model = changeRxnBounds(model, model.rxns(6),
glucosamineIN, 'u');
model = changeRxnBounds(model, model.rxns(6),
glucosamineIN, 'l');

%All 12 variables that need values

```

```

UDPgal_out = params(7)*galactoseIN;
UDPgal_to_UDPglc = (1-params(7))*galactoseIN;

if UDPgal_to_UDPglc ~= 0
    glc6P_to_glc1P = 0;
    glc6P_to_fruc6P = glucoseIN + UDPgal_to_UDPglc;
else
    glc6P_to_glc1P = params(9)*glucoseIN;
    glc6P_to_fruc6P = (1 - params(9))*glucoseIN;
end

fuc1P_to_man6P = params(10)*fucoseIN;
fruc6P_to_man6P = params(13)*glc6P_to_fruc6P;
man6P_to_fuc1P = 0;
man6P_to_fruc6P = params(11)*(mannoseIN +
fuc1P_to_man6P);
man1P_to_man6P = 0;

GDPman_to_GDPfuc = params(14)*(mannoseIN +
fruc6P_to_man6P + fuc1P_to_man6P ...
- man6P_to_fuc1P - man6P_to_fruc6P);

glcN6P = glucosamineIN + man6P_to_fruc6P +
glc6P_to_fruc6P - fruc6P_to_man6P;

[manNAc_to_glcNAc6P, UDPglcNAc_out] =
solveRecycle(params(15), params(16), glcN6P, manNAcIN);

%Now change the appropriate reaction bounds for all 12
variables identified above

model = changeRxnBounds(model, model.rxns(42), UDPgal_out
, 'u');
model = changeRxnBounds(model, model.rxns(42), UDPgal_out
, 'l');

model = changeRxnBounds(model, model.rxns(20),
UDPgal_to_UDPglc, 'u');
model = changeRxnBounds(model, model.rxns(20),
UDPgal_to_UDPglc, 'l');

model = changeRxnBounds(model, model.rxns(25),
glc6P_to_glc1P, 'u');

```

```

model = changeRxnBounds(model, model.rxns(25),
glc6P_to_glc1P , 'l');

model = changeRxnBounds(model, model.rxns(27),
glc6P_to_fruc6P , 'u');
model = changeRxnBounds(model, model.rxns(27),
glc6P_to_fruc6P , 'l');

model = changeRxnBounds(model, model.rxns(8),
fuc1P_to_man6P , 'u');
model = changeRxnBounds(model, model.rxns(8),
fuc1P_to_man6P , 'l');

model = changeRxnBounds(model, model.rxns(13),
fruc6P_to_man6P , 'u');
model = changeRxnBounds(model, model.rxns(13),
fruc6P_to_man6P , 'l');

model = changeRxnBounds(model, model.rxns(9),
man6P_to_fuc1P , 'u');
model = changeRxnBounds(model, model.rxns(9),
man6P_to_fuc1P , 'l');

model = changeRxnBounds(model, model.rxns(12),
man6P_to_fruc6P , 'u');
model = changeRxnBounds(model, model.rxns(12),
man6P_to_fruc6P , 'l');

model = changeRxnBounds(model, model.rxns(15),
man1P_to_man6P , 'u');
model = changeRxnBounds(model, model.rxns(15),
man1P_to_man6P , 'l');

model = changeRxnBounds(model, model.rxns(17),
GDPman_to_GDPfuc , 'u');
model = changeRxnBounds(model, model.rxns(17),
GDPman_to_GDPfuc , 'l');

model = changeRxnBounds(model, model.rxns(35),
manNAc_to_glcNAc6P , 'u');
model = changeRxnBounds(model, model.rxns(35),
manNAc_to_glcNAc6P , 'l');

```

```

model = changeRxnBounds(model, model.rxns(41),
UDPglcNAc_out , 'u');
model = changeRxnBounds(model, model.rxns(41),
UDPglcNAc_out , 'l');

%Lastly, removes constraints on all the outputs

for i = 44:48
    model = changeRxnBounds(model, model.rxns(i), 0 ,
'l');
    model = changeRxnBounds(model, model.rxns(i), 10000 ,
'u');
end

y = model;

end

```

C.2.15 Metabolism Analysis Mode: ECP Calibration Utility Script (File – solveRecycle.m)

```

function [ manNAc_to_glcNAc6P, UDPglcNAc_out ] =
solveRecycle( R1, R2, F1, F2 )
%Solves the mass balance associated with the recycle
stream for the model

M = [-R1 1 0 0 0; 0 0 -R2 1 0; -1 0 0 1 0; 1 -1 -1 0 0; 0
0 1 -1 -1];
b = [0; F2*R2; -F1; 0; -F2;];

x = (M^-1)*b;

manNAc_to_glcNAc6P = x(4);
UDPglcNAc_out = x(2);

end

```

C.2.16 Metabolism Analysis Mode: Print Results (File – CCM_PrintResults.m)

```
function [ output ] = CCM_PrintResults( solution, rawRxns
)
%Prints all the reaction fluxes

count = 1;
names = cell(1);
nums = zeros(1);

output = cell(length(rawRxns),2);

for i = 1:length(solution.x)

    output(i,1) = rawRxns(i);
    output(i,2) = num2cell(solution.x(i));

end

end
```