Revelation of Yin-Yang Balance in Microbial Cell Factories by Data Mining, Flux Modeling, and Metabolic Engineering

Gang Wu

Washington University in St. Louis

Follow this and additional works at: http://openscholarship.wustl.edu/eng_etds

Part of the Biodiversity Commons, Computer Sciences Commons, Engineering Commons, and the Evolution Commons

Recommended Citation
http://openscholarship.wustl.edu/eng_etds/163

This Dissertation is brought to you for free and open access by the Engineering and Applied Science at Washington University Open Scholarship. It has been accepted for inclusion in Engineering and Applied Science Theses & Dissertations by an authorized administrator of Washington University Open Scholarship. For more information, please contact digital@wumail.wustl.edu.
Revelation of ‘Yin-Yang’ Balance in Microbial Cell Factories by Data Mining, Flux Modeling, and Metabolic Engineering

by

Gang Wu

A dissertation presented to the
Graduate School of Arts & Sciences
of Washington University in
partial fulfillment of the
requirements for the degree
of Doctor of Philosophy

May 2016
Saint Louis, Missouri
Table of Contents

List of Figures ........................................................................................................ix

List of Tables........................................................................................................... xiv

Acknowledgement.................................................................................................... iii

Abstract .................................................................................................................... xvi

Chapter 1: Introduction of Fluxomics Studies and Metabolic Burden Modeling

1.1. Background of fluxomics .................................................................................. 1

1.2. A brief overview of FBA ................................................................................... 1

1.3. Introduction of $^{13}$C-MFA .............................................................................. 1

1.4. Overview of published $^{13}$C-MFA papers on bacteria species ......................... 6

1.5. Modeling work related with metabolic burden .................................................. 9

1.6. Outline of this dissertation ............................................................................... 13

1.7. Reference .......................................................................................................... 14

Chapter 2: Evaluating Physiological State of Engineered E. coli Strains by Isotopomer Constrained Flux Balance Analysis ......................................................... 22

2.1. Abstract .......................................................................................................... 22

2.2. Introduction .................................................................................................... 23

2.3. Materials and Methods ................................................................................... 26
Chapter 3: Investigate Energy Metabolism of Microbial Cell Factories by Yin-Yang Theory Research .......................................................60

3.1. Abstract......................................................................................60
3.2. Introduction................................................................................61
3.3. The energy losses in microbial cell factories ................................62
3.4. The tradeoff between product yield and energy fitness..............64
3.5. Sensitivity analysis of the energy penalty on biofuel synthesis......66
3.6 Yin-Yang theory in metabolic engineering .................................................................68

3.7. Conclusions ..................................................................................................................73

3.8. References ..................................................................................................................73

Chapter 4: Enhance Energy State of Fatty Acid Producing E. coli Strains

With *Vitreoscilla* Hemoglobin .........................................................................................87

4.1. Abstract ..........................................................................................................................87

4.2. Introduction ....................................................................................................................87

4.3. Experimental and Methods ..........................................................................................90

4.3.1. Chemicals and Strains .............................................................................................90

4.3.2. Plasmid construction ...............................................................................................91

4.3.3. Medium and culture conditions ...............................................................................92

4.3.4. Fatty acid measurement ..........................................................................................92

4.3.5. Simulate cellular physiology with flux balance model .............................................93

4.4. Results and discussions ...............................................................................................94

4.4.1. Growth kinetics and fatty acid production ..............................................................99

4.4.2. Expression of VHb affects the degree of unsaturation of free fatty acid ...................95

4.4.3. Effect of oxygen and maintenance energy on fatty acid production .......................96

4.5. Conclusions ..................................................................................................................98

4.6. References ..................................................................................................................98
Chapter 5: Build Web-Based Platform for Fluxomics Studies: MicrobesFlux

Rebuild and Website Development ................................................................. 112

5.1. Abstract ........................................................................................................... 112
5.2. Introduction ..................................................................................................... 113
5.3. Implementation ............................................................................................... 115
  5.3.1. MicrobesFlux update ................................................................................ 115
  5.3.2. New features of reloaded MicrobesFlux .................................................. 116
  5.3.3. Development of websites for fluxomics studies ....................................... 116
5.4. Results ........................................................................................................... 116
5.5. Availability and requirements ....................................................................... 117
5.6. References ...................................................................................................... 118

Chapter 6: Rapid Prediction of Bacterial Fluxomics Using Machine Learning and Constraint Programming ................................................................. 125

6.1. Abstract ........................................................................................................... 125
6.2. Authors’ Summary .......................................................................................... 126
6.3. Introduction ..................................................................................................... 126
6.4. Methods .......................................................................................................... 129
  6.4.1. Data collection and preprocessing ............................................................ 129
  6.4.2. Feature selection and scaling ..................................................................... 130
  6.4.3. Machine learning algorithm selection ....................................................... 131
7.3. Methods ...................................................................................................................... 169
7.3.1. Database availability and record structure................................................................. 169
7.3.2. Text mining methods ................................................................................................. 170
7.3.3. Fast search via BigQuery ......................................................................................... 171
7.3.4. Data analysis ........................................................................................................... 172
7.4. Results and discussions ............................................................................................... 173
7.4.1. Most related words of ‘metabolic engineering’, ‘environmental engineering’, ‘synthetic
biology’, ‘systems biology’, and ‘metabolic flux’ ............................................................. 173
7.4.2. Compare the difference and similarity between two different terms ...................... 174
7.4.3. Identify the developing trend of a specific term ......................................................... 175
7.4.4. Advantages and Limitations of Big Data workflow .................................................. 177
7.5. Conclusions ................................................................................................................ 178
7.6. References .................................................................................................................. 178

Chapter 8: Conclusions And Future Perspectives ......................................................... 192

8.1. Conclusions ................................................................................................................ 192
8.2. Future directions to solve intracellular energy crisis .................................................... 194
8.3. Personal views on the future of microbial cell factories .............................................. 198
8.4. References .................................................................................................................. 199

9. Appendix ....................................................................................................................... 204

9.1. Appendix I .................................................................................................................. 204
9.2. Appendix II ......................................................................................................................206

10. Side Projects ......................................................................................................................276

10.1. Phytotoxicity of metal oxide nanoparticle (NP) on plant seeds ........................................276

10.2. NP electrospray facilitates seed germination ....................................................................388

Vita ........................................................................................................................................322
List of Figures

Figure 1.1 Procedure of GSM reconstruction and FBA .................................................. 3

Figure 1.2 A general procedure of $^{13}$C-MFA .......................................................... 5

Figure 1.3 Percentage of $^{13}$C-MFA paper on each bacteria species .......................... 7

Figure 1.4 Number of $^{13}$C-MFA papers published on each journal ............................ 8

Figure 2.1 Diagram of $^{13}$C flux as constraints for FBA ............................................ 48

Figure 2.2 Central metabolic flux of each strain determined by $^{13}$C-MFA .................. 49

Figure 2.3a-b Energy analysis of four strains at different energy conditions ................. 50

Figure 2.4 Evolutionary fitness of four strains in this study ......................................... 54

Figure 2.5a-e: Influence of P/O ratio and maintenance energy on isobutanol production potential (growth rate for strain 1, JCL260), simulated by FBA ................................. 54

Figure 2.6a-e: Influence of oxygen uptake flux and maintenance energy on isobutanol production potential (growth rate for strain 1, JCL260), simulated by FBA ...................... 57

Figure 3.1 Cell carbon and energy metabolism illustrated by Yin-Yang Theory .......... 80

Figure 3.2a-d: Genome-scale FBA models for microbial biofuel mole-carbon yields from glucose ........................................................................................................ 81

Figure 3.3 Energy fitness and productivities in microbial cell factories .......................... 84
Figure 3.3a: The trend of metabolic entropy changes (unit: ATP generation per glucose). In optimal metabolism, one mole of glucose generates 38 ATP for biosynthesis. Under constraints of P/O ratios and maintenance loss, less ATP can be generated (i.e., increase of metabolic entropy).

Figure 3.3b: The transition from carbon limitation to energy limitation with the increase of product yield. In many cases, the energy limitation prevents strains from achieving the yield and titer at break-even point.

Figure 3.3c: Cascade of energy changes (Heat of combustion) during biofuel synthesis from glucose.

Figure 4.1 Structure of Vitreoscilla hemoglobin in active dimer form.

Figure 4.2 Genetic manipulations to insert VHb into pA58c-TR.

Figure 4.3 Optimization of IPTG concentrations for VHb50 expression.

Figure 4.4 Growth curve for three fatty acid producing strains.

Figure 4.5 Fatty acid productions after 24 hr of IPTG Induction.

Figure 4.6 Free fatty acid production profiles for control strain and VHb strain after (a) 8 hr and (b) 24 hr of IPTG induction.

Figure 4.7 Effect of oxygen flux and maintenance energy on fatty acid yield at (a) exponential phase and (b) late exponential phase.

Figure 5.1 $^{13}$C-MFA protocol and sources of flux analysis variance.
Figure 5.2 The webpage of our platform for comprehensive fluxomics studies (http://fluxomics.net).............................................................................................................................122

Figure 5.3 The webpage of WUFlux (http://13cmfa.org), which can be accessed and freely download..........................................................................................................................................................................................................................................................123

Figure 5.4 The webpage of Amazon server EC2, (a) all Amazon web services, (b) buckets of our websites ...........................................................................................................................................................................................................................................................................124

Figure 6.1 A universal central metabolic pathway for bacteria: The central carbon metabolic pathway is simplified into 29 fluxes in MFlux...........................................................................................................................................................................................................................................................................157

Figure 6.2 Statistical analysis of central metabolic fluxes collected in our database. “Flux range” represents variations of each fluxes among $^{13}$C-MFA database; “95% confidence interval” represents 95% of flux data were within a small range; “Average flux value” are the mean of flux values from $^{13}$C-MFA database...........................................................................................................................................................................................................................................................................158

Figure 6.3 A flow chart of MFlux algorithm. This diagram is to illustrate the detailed procedures for our algorithm. ...........................................................................................................................................................................................................................................................................159

Figure 6.4 A comparison of three different algorithms: SVM, kNN, and decision tree: The best cross-validation results on 29 fluxes are compared. All tests in this step were performed only on the WT database...........................................................................................................................................................................................................................................................................160

Figure 6.5 Best results by SVM for WT and WP databases. Both the linear and the RBF kernels are considered in a grid search, and the results from WP database is much better than from the WT database...........................................................................................................................................................................................................................................................................161
Figure 6.6 A comparison between the linear kernel and the RBF kernel for SVM. The results are quite similar.................................................................162

Figure 6.7 Summary of root mean squared error (RMSE) from 20 case studies: averaged flux from $^{13}$C-MFA database; machine learning, and MFlux. The average RMSE is 7.7 from machine learning alone and 5.6 from MFlux.................................................................163

Figure 6.8 A comparison of $^{13}$C-MFA, the flux predicted by ML, and the flux predicted by MFlux in case 8. *B. subtilis* was incubated in a shake flask (37 °C, 300 rpm, aerobic condition), and supplied with labeled succinate and glutamate as carbon sources in M9 minimal medium.................................................................164

Figure 6.9 A comparison of $^{13}$C-MFA, the flux predicted by ML flux, and the flux predicted by MFlux. *G. thermoglucosidasius* M10EXG was incubated in sealed bottles (micro-aerobic condition), supplied with glucose as a carbon source. RMSE$_{ML} =$ 4.0, RMSE$_{MFlux} =$ 3.0.................................................................165

Figure 6.10 A comparison of $^{13}$C-MFA, MFlux and the flux predicted by FBA. FBA Analysis is simulated by *E. coli* iJO1366 model with defaulted boundary settings from the reference (Orth *et al*. 2011). (A) *E.coli* fluxome of glucose metabolism was precisely measured via parallel labeling experiments (a recent paper not in our database) (Crown *et al*. 2015). RMSE$_{FBA} =$ 11.3, RMSE$_{MFlux} =$ 6.5. (B) *E. coli* fluxome of glycerol and glucose co-metabolism were measured by Dr. Yao and Dr. Shimizu (unpublished data). *E. coli* strain was cultured in chemostat fermentor with a working volume of 1 L (37 °C). The dilution rates in the continuous culture were 0.35 h$^{-1}$. $[1-^{13}$C$]$ glucose and $[1, 3-^{13}$C$]$ glycerol were used for tracer experiments. The flux calculation is
based on previous method (Fong et al. 2006; Peng et al. 2004). RMSE_{FBA} = 22.5, RMSE_{MFlux} = 5.1

Figure 7.1a Total number of journals in the database at different time
Figure 7.1b Total number of papers in the database increase at different time
Figure 7.2 Structure of nxml file
Figure 7.3 Word cloud of ‘metabolic engineering’
Figure 7.4 Word cloud of ‘environmental engineering’
Figure 7.5 Word cloud of ‘synthetic biology’
Figure 7.6 Word cloud of ‘systems biology’
Figure 7.7 Word cloud of ‘metabolic flux’
List of Tables

Table 2.1 Detailed information of plasmids and strains used in this study…………………………..44

Table 2.2 Physiological information of strains used in this study.................................................44

Table 2.3 Metabolic network for 13C-MFA calculation.................................................................46

Table 2.4 Energy metabolism of different strains.................................................................47

Table 4.1 DNA sequence of all primers used in this work.........................................................105

Table 6.1 Summary of 20 cases of study.....................................................................................164

Table 7.1 Comparison of ‘metabolic engineering’ and ‘synthetic biology’, similarity
69.6%...............................................................................................................................................182

Table 7.2 Development trend of ‘metabolic engineering’ during 2000 ~ 2009 and 2010 ~ 2015,
similarity 81%.................................................................................................................................183

Table 7.3 Development trend of ‘biofuel’ during 2000 ~ 2009, 2010 ~ 2015, and 2015 similarity
72.4% and 88%................................................................................................................................184
Acknowledgements

During the past four years, I would like to convey my sincere gratitude to my adviser, Dr. Yinjie J. Tang, for bringing me into this PhD program, teaching me so much and providing tremendous supports on my PhD research work. I would also like to express my appreciation to all the members in my committee: Dr. Pratim Biswas, Dr. Cynthia Lo, Dr. Tae Seok Moon, Dr. Zi Chen, Dr. Ryan Senger, and Dr. Venkat Subramanian (previous committee member) for their helpful suggestions and guidance.

Many thanks go to lab colleagues in our group, including Lian He, Whitney Hollinshead, Mary Abernathy, Tolutola Oyetunde, alumni Dr. Le You, Dr. Arul Varman, Dr. Yi Xiao, and visiting Professors, Dr. Yi Yu, Dr. Haifeng Hang, Dr. Lifeng Peng, for their countless assistances in research and friendship. Thanks also extend to Patty Wurm for laboratory supports, Tim McHugh for helping with remote computing and video. I also would like to thank all staffs in EECE department, who make the department a wonderful place to work and study. Further, I would like to thank James Ballard at writing center of engineering school for help in my writings.

Besides, I would like to thank my colleague Di Liu, Wen Jiang for helping the VHb experiments, Dr. Forrest Sheng Bao at University of Akron for numerous discussions and helps for ThunderFlux project, and Dr. Eric You Xu for many insightful suggestions for MicrobesFlux reconstruction. My sincere thanks go to Dr. Shi Chen, Dr. Cristina Lanzas, Dr. Alison Buchan, and Dr. Louis Gross at the University of Tennessee for their encouragements during my visit at NIMBioS, and to Dr. Yi Sun, Dr. Yang Liu, Dr. Qunyuan Zhang, and Dr. Fei Zhong for their helps during my summer intern at Sigma-Aldrich.
Also, I want to express my deepest gratitude to my parents and my wife, for their unending supports, encouragements, and love during my PhD study. Last but most important, nothing can be realized without the grace of the ONE, who saved me from the Death Valley, changed me through nine years of graduate life. All glory goes to Him.

Gang Wu

Nov 2015
ABSTRACT OF THE DISSERTATION

Decipher ‘Yin-Yang’ Balance of Microbial Cell Factories by Data Mining, Flux Modeling, and Metabolic Engineering

By

Gang Wu

Doctor of Philosophy in Energy, Environmental, and Chemical Engineering

Washington University in St. Louis, 2015

Professor Yinjie Tang, Chair

The long-held assumption of never-ending rapid growth in biotechnology and especially in synthetic biology has been recently questioned, due to lack of substantial return of investment. One of the main reasons for failures in synthetic biology and metabolic engineering is the metabolic burdens that result in resource losses. Metabolic burden is defined as the portion of a host cell’s resources — either energy molecules (e.g., NADH, NADPH and ATP) or carbon building blocks (e.g., amino acids) — that is used to maintain the engineered components (e.g., pathways). As a result, the effectiveness of synthetic biology tools heavily depends on cell capability to carry on the metabolic burden. Although genetic modifications can effectively engineer cells and redirect carbon fluxes toward diverse products, insufficient cell ATP powerhouse is limited to support diverse microbial activities including product synthesis. Here, I employ an ancient Chinese philosophy (Yin-Yang) to describe two contrary forces that are interconnected and interdependent, where Yin represents energy metabolism in the form of ATP, and Yang represents carbon metabolism. To decipher “Yin-Yang” balance and its implication to microbial cell factories, this dissertation applied metabolic engineering, flux analysis, data
mining tools to reveal cell physiological responses under different genetic and environmental conditions.

Firstly, a combined approach of FBA and $^{13}$C-MFA was employed to investigate several engineered isobutanol-producing strains and examine their carbon and energy metabolism. The result indicated isobutanol overproduction strongly competed for biomass building blocks and thus the addition of nutrients (yeast extract) to support cell growth is essential for high yield of isobutanol. Based on the analysis of isobutanol production, 'Yin-Yang' theory has been proposed to illustrate the importance of carbon and energy balance in engineered strains. The effects of metabolic burden and respiration efficiency (P/O ratio) on biofuel product were determined by FBA simulation. The discovery of ‘energy cliff’ explained failures in bioprocess scale-ups. The simulation also predicted that fatty acid production is more sensitive to P/O ratio change than alcohol production. Based on that prediction, fatty acid producing strains have been engineered with the insertion of *Vitreoscilla* hemoglobin (VHb), to overcome the intracellular energy limitation by improving its oxygen uptake and respiration efficiency. The result confirmed our hypothesis and different level of trade-off between the burden and the benefit from various introduced genetic components. On the other side, a series of computational tools have been developed to accelerate the application of fluxomics research. Microbesflux has been rebuilt, upgraded, and moved to a commercial server. A platform for fluxomics study as well as an open source $^{13}$C-MFA tool (WUFlux) has been developed. Further, a computational platform that integrates machine learning, logic programming, and constrained programming together has been developed. This platform gives fast predictions of microbial central metabolism with decent accuracy. Lastly, a framework has been built to integrate Big Data technology and text mining to interpret concepts and technology trends based on the literature survey. Case studies have been
performed, and informative results have been obtained through this Big Data framework within five minutes.

In summary, $^{13}$C-MFA and flux balance analysis are only tools to quantify cell energy and carbon metabolism (i.e., Yin-Yang Balance), leading to the rational design of robust high-producing microbial cell factories. Developing advanced computational tools will facilitate the application of fluxomics research and literature analysis.
CHAPTER ONE

INTRODUCTION OF FLUXOMICS STUDIES AND METABOLIC BURDEN MODELING

1.1. Background of fluxomics

Systems biology reveals intricate cellular metabolic and regulatory activities by a series of high-throughput methods. Development and application of those high-throughput methods raise up their respective research field defined as ‘omics’, including genomics (sequencing and annotation of genomic DNA), transcriptomics (determination of global gene transcriptional level), proteomics (characterization of structure and function of individual protein), metabolomics (assay of metabolite profile), and fluxomics (infer rate of each single biochemical reaction within metabolic network) (Tang et al. 2009a).

In the realm of fluxomics, $^{13}$C Metabolic Flux Analysis ($^{13}$C-MFA) and Flux Balance Analysis (FBA) are basic quantitative approaches to unveil activities of metabolic reactions. Both approaches are built upon many assumptions, for instance, steady state or quasi-steady state assumption (which indicates there is no net flux of those intermediates, and the sum of input fluxes equals to sum of output fluxes), homogenous cell culture assumption (local environment and metabolic state of each cell is considered to be equally same). During the exponential growth phase of a well cultivated single cell organism culture, these assumptions are reasonable for most times and modeling calculations based on them seem relatively accurate.

1.2. A brief overview of FBA
FBA is a bio-mathematical approach to calculate metabolic flux profiles and has been extensively developed during recent two decades. By building up genome-scale metabolic network model, FBA is able to calculate large scale models with more than 2000 reactions which include over 30% of genes of the whole genome. (Orth et al. 2011; Monk et al. 2013) As a powerful mathematical modeling tool, FBA is able to make predictions on growth rate, product yield, nutrient requirements, physiology of knockout phenotype, track extreme pathways, and guide rational metabolic engineering based on only a few inputs. (Kauffman et al. 2003; Edwards et al. 2002; Orth et al. 2010a; Becker et al. 2007; Bordbar et al. 2014) Therefore, it became popular and widely accepted by researchers of diverse fields (Tang et al. 2009a). Until now, Genome scale model (GSM) reconstruction has been performed for more than one hundred species (http://systemsbiology.ucsd.edu/InSilicoOrganisms/OtherOrganisms) including bacteria, eukaryotic, and archaeal species and this number is still increasing now (Orth et al. 2010a). Meanwhile, many efforts have been made to integrate FBA model with other omics data (e.g., transcriptomics, proteomics, and metabolomics) (Gowen and Fong 2010; Schellenberger et al. 2011; Åkesson et al. 2004; Coquin et al. 2008; Winter and Krömer 2013).

A general procedure for GSM reconstruction and FBA is depicted in Figure 1.1.: first collecting information from pathway database and identify existing genes and pathways, then performing gap filling and patching the gaps among metabolic network of GSM. The stoichiometry matrix of metabolic reactions and list of metabolites will be extracted out subsequently. Constrained linear programming is carried out based on the guidance of objective function and flux profile will be obtained. Maximization of biomass growth is the most common objective function and has been proven to be reasonable in many case studies, especially for those microbes in exponential growth phase (Orth et al. 2010a; Khannapho et al. 2008).
Calculation of FBA is a LP (Linear Programming) process. In general, there are much more fluxes than metabolites in FBA, resulting in large freedom of solution space and underdetermined system (Vallino and Stephanopoulos 1993). Even with a defined objective function, a series of constraints (e.g., thermodynamic properties, exchange/overflow fluxes, pH balance etc) and other information (e.g., biomass composition) are essential for meaningful simulation results of cellular metabolism. Among those constraints, overflow measurements are regularly used to define FBA boundaries. Precise measurements of some overflows require researchers developing professional skills in instruments such as HPLC. Similarly, biomass composition determination is expensive and labor intensive (Tang et al. 2009a). Recently, lots of efforts have been made to develop effective constraints to relieve the tiresome work of measurement, including addition of flux ratio on critical nodes (FBrAtio) (Yen et al. 2013;
McAnulty et al. 2012), linking with other omics data such as RNAseq or microarray data (as mentioned above), specific proton flux (SPF) (Senger and Papoutsakis 2008), enzyme activity assay, or even the flux values calculated from $^{13}$C-MFA results (Blank et al. 2005).

1.3. Introduction of $^{13}$C-MFA

As an alternative tool for metabolic flux analysis, $^{13}$C-MFA adopts the information from labeling experiments as the constraints for modeling calculation and takes a different procedure to obtain the flux profile (Shown in Figure 1.2.). $^{13}$C labeling experiment is an important part in $^{13}$C-MFA. Labeled substrates are added into minimal medium to feed cells, and biomass is harvested when a metabolic & isotopic steady state is reached. Harvested biomass is pretreated and derivatized for further analysis (NMR or GC-MS). Proteinogenic amino acids can be analyzed via GC-MS measurement following a well-developed protocol (Fischer and Sauer 2003; Zamboni et al. 2009). The advantages of proteinogenic amino acids approach lie in its easy manipulation, fast process, relatively high accuracy, high robustness and relatively low requirements on instruments. Therefore, it can be potentially developed into an automatic process (e.g., use robot). Using labeling information from central metabolites requires different derivatization methods (You et al. 2014). Some unstable metabolites requires fast quenching method, and further analysis of many metabolites needs LC MS-MS (Young et al. 2011), which provides more analytic power yet costs much more than GC-MS.
Normally mass isotopomer distribution data from 16 amino acids can be used for flux profile calculation (Tang et al. 2009a). The objective function of $^{13}$C-MFA is to minimize error between simulated and experimental determined mass isotopomer distribution of amino acids, which can be expressed as equation 1. The computational methods for $^{13}$C-MFA include the cumomer method (Möllney et al. 1999), isotopomer path tracing method (Forbes et al. 2001), and most recently, the Elementary Metabolic Unit (EMU) method (Antoniewicz et al. 2007a). Among them, EMU uses the minimum set of information to track atom transition among central metabolism, and is proven to be very efficient (Young et al. 2008).
13C-MFA has proven its competence in finding new pathway (Tang et al. 2007a), validation of gene functions (Tang et al. 2007a), medium design (Zhuang et al. 2011a), and profiling metabolism of engineered strain (He et al. 2014; Becker et al. 2011). However, there are only a few cases with the aid of 13C-MFA can improve yield of desired product, (Tang et al. 2012), and 13C-MFA is employed as a powerful tool of validation rather than prediction. Furthermore, it is still difficult to quickly determine flux profiles of strain library (hundreds of strains) to know the variances among different phenotypes at the flux level.

A series of computational tools have been developed in the field of fluxomics. Among them, COBRA toolbox developed by Palsson’s group at UCSD is the most famous software (Becker et al. 2007; Schellenberger et al. 2011). COBRA is capable in many functions, including both FBA and 13C- MFA. In the field of 13C-MFA, published modeling framework includes 13C-FLUX and 13C-FLUX2 by Wiechert group (Wiechert et al. 2001; Weitzel et al. 2013), FiatFlux by Zamboni group (Zamboni et al. 2005a), Metran by Antoniewicz group, and INCA by Young group (Young 2014). All those software tools have greatly promote researches in the fluxomics field.

1.4. Overview of published 13C-MFA papers on prokaryotic species

We collected most 13C-MFA papers on bacteria species published during the past twenty years (by Dec 2014). Through a brief survey, we found some important facts about 13C-MFA research:

(1) Fact of microbial species that 13C-MFA papers worked on: Most 13C-MFA papers are focusing on three model species: E. coli, B. subtilis, and C. glutamicum, which cover nearly 70% of our paper collections (shown in Figure 1.3.). This can be explained by that genetic
manipulations are so mature for those three model species, that many mutants and engineered strains have been created by researchers around the world. Also, there are lots of reports on metabolic network and biomass composition for those model species, researchers don’t need to spend time or money on those experiments. For the rest 30% papers, they are on pathogenic species, environmental essential species, and chemical or fuel potential producers.

![Pie chart showing the percentage of 13C-MFA papers on each bacteria species](image)

**Figure 1.3 Percentage of 13C-MFA paper on each bacteria species**

(2) Fact of scientific journals 13C-MFA papers published on: the top three journals that 13C-MFA published are ‘Metabolic Engineering’, ‘Applied and Environmental Microbiology’, and ‘Biotechnology and Bioengineering’ (as shown in Figure 1.4.). This result is very informative: people employed 13C-MFA as a complimentary tool of metabolic engineering and 13C-MFA do provide quantitive information of central carbon metabolism.
(3) Fact of researchers active in $^{13}$C-MFA field: the top three researchers published most $^{13}$C-MFA papers are Dr. Uwe Sauer (mainly work on *Bacillus subtilis* and *Escherichia coli*), Dr. Christoph Wittmann (mainly work on *Corynebacterium glutamicum*), and Dr. Kazuyuki Shimizu (mainly work on *Escherichia coli*).

1.5. Modeling work related with metabolic burden

Metabolic burden was first defined as ‘expression of foreign proteins utilize a significant amount of the host cell’s resources, removing those resources away from host cell metabolism and placing a metabolic load (or burden) on the host’ by Glick (Glick 1995). With product scope of genetic modifications extended to various chemicals; we also extend the concept of metabolic
burden to all cellular energy not used for biomass synthesis or product formation. There are six major sources of metabolic burden:

(a) Defense of internal stress (e.g., imbalanced NADH/NADPH or NAD/NADH ratio, excess proton). A famous example is that isobutanol production leads to imbalance of cofactor utilization in *E.coli* (Bastian *et al.* 2011).

(b) Defense of environmental stress (e.g., toxic compounds, O$_2$ stress). A good instance is that the fluxome of *Shewanella oneidensis* show robustness under salt stress (Tang *et al.* 2009b).

(c) Cost of protein overexpression and plasmid maintenance (including the cost for turnover and protein incorrect folding). For instance, protein overexpression significantly boosted fluxes of the TCA cycle and acetate overflow (Heyland *et al.* 2011).

(d) Defense of specific stress (toxicity) from metabolites or enzymes. A typical example is that overproduction of fatty acid will cause severe stress on cell membrane (Lennen *et al.* 2011).

(e) Energy spilling reactions (e.g., futile cycle). Cells just waste the energy when the energy source is sufficient (Hoehler and Jorgensen 2013a; Russell and Cook 1995).

(f) Energy for cell mobility, which only costs for 2% of total energy (Russell and Cook 1995).

Burden from protein expression has been noticed even over fifty years ago: Induced *E. coli* cells demonstrated a significant decrease in growth rate compared with uninduced cells (Novick and Weiner 1957). With the advance of genetic manipulation technology (e.g., restriction enzyme, DNA sequencing method) and successful commercialization of recombinant protein (e.g., insulin by Genentech) during late 1970s and early 1980s, overexpression of various proteins in order to get desired products has been attempted. The negative effects of protein and
plasmid burden were realized by a broad range of researchers (Schaaff et al. 1989; Jensen et al. 1993; Birnbaum and Bailey 1991). To better understand and simulate the impacts of metabolic burdens, a series of models were proposed or began to include the effect of burden into consideration. Ollis and Chang took the factor of plasmid instability into unstructured kinetic models; and their model is able to predict the effect of different inoculation ratio on final product formation (Ollis and Chang 1982). Lee and Bailey developed a model include the mathematical description of plasmid replication. And their model simulation results well matched the experimental observations that increased plasmid content leads to decrease growth rate. Further, the model also predicted the maximal intracellular product accumulation with respect to growth rate, which was simultaneously verified by experimental data.(Lee and Bailey 1984) In another work authored by Bailey, an empirical parabolic relationship was adopted to integrate with Monod equation to investigate the effects of different plasmid copies and various medium composition on beta-lactamase specific activity (Seo and Bailey 1985). In a later work, Bailey employed a structure model mathematically describing the competition between chromosomal- and plasmid-based expression system for cellular resource (e.g., transcription and translation machinery). This model well predicted the influences of different promoters and RBS strength on growth rate. In addition, the simulation result also revealed that the capability of intracellular transcriptional machinery was a limiting factor of heterogeneous gene expression (Peretti and Bailey 1987).

Other models rather than kinetic or structured model were also employed to simulate the effects of metabolic burden. For instance, Snoep et al. took the factor of protein burden into the metabolic control model by assigning a coefficient. Their model was able to appropriately explain the effects of glycolytic enzymes overexpression in Zymomonas mobilis (Snoep et al.
An empirical model was also proposed to interpret protein heterologous expression in *E. coli*. This model was experimentally verified and the authors identified ribosome allocation as the limiting factor of growth during this process (Carrera *et al.* 2011; Somerville *et al.* 1994). A recent paper proposed a mechanistic model which considered three trade-offs on cellular resource (i.e., total protein, free ribosomes, and cellular energy) (Weiße *et al.* 2015). This model predicted well on cellular behaviors and the interaction between a synthetic circuit and its host. All those models provide reasonable quantifications of the effects by various factors related with the metabolic burden on cellular metabolism. However, it is still a challenge to apply those models (with their respective parameter set) directly into another system without extensive efforts on experiments and data fitting.

Flux balance model potentially provides an alternative strategy to quantify the metabolic burden. Weber *et al.* first employed FBA to investigate the effects of recombinant protein production on flux distribution and growth. (Weber *et al.* 2002) In their work, the amino acid composition of human basic fibroblast growth factor (hFGF-2) was considered and calculated as a specific flux. With maximization of biomass growth as the objective function, FBA successfully predicted that increased energy demand was satisfied by up-regulation of fluxes within EMP pathway and TCA cycle, as well as activated transhydrogenase flux. Later, Özkan *et al.* included the effect of heterogeneous plasmid maintenance and antibiotics marker protein expression into FBA model to simulate the heterogeneous protein expression induced by IPTG in a minimal medium supplied with amino acids (Özkan *et al.* 2005). The authors claimed that the predicted relative flux distribution was in the same trend as reported expression profiles; however, there are some apparent variations between prediction and real values (e.g., P/O ratio).
Besides, the researchers didn’t set any boundary of amino acid uptake fluxes, which might lead to further variations.

\(^{13}\)C-MFA was also employed to interpret the metabolic burden of plasmid maintenance. (Wang et al. 2006a) The authors found that the strain hosting high copy plasmid had significantly lower relative flux in TCA cycle and higher flux in acetate secretion and ATP maintenance. The energy metabolism of engineered strains producing biofuel was also investigated by using \(^{13}\)C-MFA. And the researchers found that the maintenance energy of fatty acid producing strain is two-fold of maintenance energy in the control strain (He et al. 2014). And this result partially explained numerous failures of biofuel production scale-up projects: high maintenance requirement leads to strain instability. To further explore the metabolism of strains under industrial fermentation (normally use complex medium rather than minimal medium), dynamic flux analysis and fast quenching method are required to resolve the metabolic flux and energy metabolism (Antoniewicz et al. 2007b; Zamboni 2011).

1.6. Outline of this dissertation

In Chapter two, we integrated FBA and \(^{13}\)C-MFA to investigate the metabolism of several isobutanol-producing strains. In particular, the energy metabolism, and several related factors such as the P/O ratio, the oxygen condition, and the maintenance energy were investigated.

In Chapter three, after reviewing recent successes and failures of metabolic engineering projects, we proposed the Yin-Yang theory of metabolic engineering: carbon metabolism and energy metabolic within microbial cell factories should be balanced. We employed FBA to predict the effects of maintenance energy (metabolic burden) and P/O ratio on biofuel yields in \(E.\ coli\). We also provided several strategies to solve the energy bottleneck in engineered strains.
In Chapter four, we attempted to solve the energy bottleneck in fatty acid producing strain by insertion of *vhb* gene, which facilitates oxygen uptake of the host cell. We tested three VHB variants with different oxygen transfer capabilities. Compared with control strain, engineered strain with wild-type VHB only showed a decreased growth as well as reduced fatty acid production because genetic modification brought more metabolic burden than benefits; while strain with VHB50 demonstrated higher cell density, as well as increased fatty acid accumulation.

In Chapter five, we developed a series of modeling tools for fluxomics studies. First, we rebuilt MicrobesFlux on a commercial server (Amazon AWS) to make the systems more usable. Second, we also developed an open source $^{13}$C-MFA tool (WUFlux) in MATLAB. Third, we designed and developed a web-based platform, to make all our fluxomics tools freely accessed and downloaded through the Internet.

In Chapter six, we collected $^{13}$C-MFA data from published literature. Based on that information, we developed a web-based computational platform (MFlux) that directly predicts bacterial central metabolism via machine learning, constraint programming, and quadratic programming. We performed cases studies with our platform and compared with FBA predictions. The results indicated that MFlux can yield decent results close to $^{13}$C-MFA values, and better than FBA predictions.

In Chapter seven, we developed a platform providing fast literature analysis by using text mining and Big Data technology. We performed several case studies to demonstrate its functionality: (a) display word cloud of a specific term; (b) compare difference between different terms; (c) show the developing trend and current status of a specific term.
In Chapter eight, we summarized all projects in this dissertation and provided personal suggestions for the future directions.

1.7. Reference


CHAPTER TWO

EVALUATING PHYSIOLOGICAL STATE OF ENGINEERED *E. COLI* STRAINS BY ISOTOPOMER CONSTRAINED FLUX BALANCE ANALYSIS

2.1. Abstract

Metabolic engineering, especially the introduction of exogenous plasmids into the cell, imposes considerable burdens on cell physiology. For example, plasmid replication, protein overexpression and metabolite accumulation significantly affect the growth rate, expression of native proteins, energetic metabolism, and cell composition of the host cell. Furthermore, biosynthesis of products may result in severe metabolic stresses and cause deleterious impact on both the cell membrane and the energy metabolism. This study aims to understand the metabolic shifts in engineered microbial hosts. Specifically, we have integrated $^{13}$C-MFA and genome-scale FBA (flux balance analysis) to investigate the physiologies of engineered *E.coli* strains for isobutanol production.

On the experimental side, we performed labeling experiments on several engineered *E. coli* strains (high performance JCL260 strain from James Liao Lab, low performance BW25113 wild type strain). Under aerobic growth conditions, we measured both the strain’s growth and isotopomer data of their key proteinogenic amino acids. Subsequently, we used a $^{13}$C-MFA ($^{13}$C-metabolic flux analysis) model to profile the central metabolism based on the isotopomer data. $^{13}$C-MFA could only determine scratchy ranges of fluxes in central metabolism. To obtain a broad metabolic solution, we built a large-scale flux balance analysis (FBA) model, which is constrained by $^{13}$C-MFA results. The integrated FBA model relied on objective functions to
evaluate flux distributions. In addition, we tested the sensitivity of the model prediction towards changes in the energy metabolism (ATP maintenance and P/O ratios) and biomass composition equations. By extensively comparing the fluxomics results between the engineered strains, we discovered several metabolic features of the high performance JCL260. First, the JCL260 strain could up-regulate its NADPH production pathways and minimize its overflow metabolism. Second, P/O ratios have relatively a small impact on its optimal isobutanol yield. Third, isobutanol overproduction strongly competes for biomass building blocks and thus addition of nutrients (yeast extract) to support cell growth is essential for high yield of isobutanol. Finally, model sensitivity analysis also implied that isobutanol production is very sensitive to the metabolic burden. Furthermore, isobutanol production pathway is less susceptible to oxygen limitation therefore more likely to achieve high yield compared with biodiesel.

**Key words:** isobutanol, $^{13}$C-MFA, P/O ratio, maintenance energy, production yield

**2.2. Introduction**

Metabolic engineering aims to get desired products through reshaping metabolic network with the aid of DNA recombination technology (Bailey 1991). In general, exogenous plasmids are introduced into host strains over-expressing enzymes to pull flux into related pathways; while chromosomal genes encoding bypass fluxes are knocked out to reduce flux competition. Genetic modifications, especially the presence of plasmids inside cell, impose a considerable influence on cell physiology. Metabolic burdens contributed by protein expression and plasmid replication significantly affect growth rate, expression of native proteins, energetic metabolism, and cell composition of host cells (Birnbaum and Bailey 1991; Özkan *et al.* 2005; Rozkov *et al.* 2004). Furthermore, generation of certain metabolites such as alcohol or fatty acid results in severe
oxidative stress *in vivo*, damages on cell membrane as well as on transport system and energy metabolism (Lennen et al. 2011; Nicolaou et al. 2010; 2009). All those factors enforce host cells to adjust their metabolism in response to burdens and stress (He et al. 2014). Quantitive understandings over such shifts would be beneficial for a deeper view of cellular regulation, and may shed light on strain rational design for the sake of metabolic engineering and bioprocess scale-up (Sauer et al. 1998; Garcia-Ochoa and Gomez 2009).

Fluxomics, the functional systems biology tool, have been employed to investigate the physiological alternations of engineered strains quantitively (Vallino and Stephanopoulos 1993; Antoniewicz et al. 2007b). As the two basic approaches in the realm of fluxomics, $^{13}$C-MFA ($^{13}$C metabolic flux analysis) measures *in vivo* flux information mainly based on mass isotopomer distribution of amino acids (isotopic fingerprint), while FBA (Flux Balance Analysis) predicts flux distribution in genome-wide metabolic network under presumed objective functions, which describe the “possible” metabolic potential of microbial hosts (Orth et al. 2010a; Stephanopoulos 1999). $^{13}$C-MFA is able to precisely determine flux profile, but limited its scope to central carbon metabolic pathways (Chen et al. 2011). FBA has been widely employed to interpret metabolism of a variety of species at the genome-scale, to identify gene essentiality, and to reveal the trend of adaptive evolution (Feist et al. 2009; Ibarra et al. 2002; Famili et al. 2003). Accurate quantification of cellular metabolism is highly dependent on the selection of appropriate objective function for FBA. Maximizing biomass growth is the most common objective function and it works well for cells in exponential growth phase (Schuetz et al. 2007; Knorr et al. 2007). However, in many cases, cell does not behave in the manner of any optimal strategies, which raise a challenge for choosing an appropriate objective function (Schuetz et al. 2007; Schuetz et al. 2012). On the other hand, introduction of a series of constraints in the process of linear
programming would greatly reduce the solution space; therefore enable more accurate predictions on flux profile. Constraints such as overflow flux, energy balance, membrane site occupancy, and proton gradient represent the laws of thermodynamics, physics and physiology working on cellular metabolism (Senger and Papoutsakis 2008; Beard et al. 2002; Zhuang et al. 2011b). The idea to combine the advantages of both approaches together as synergistic tools to overcome shortcomings of individual has been attempted in distinctive ways by various groups (Blank et al. 2005; Chen et al. 2011).

To decipher physiological conditions of engineered strains by fluxomics tools, several challenges have to be seriously circumvented. The first challenge is regarding the ATP generation by oxidative phosphorylation: Theoretically, the ideal maximal P/O ratio is 2 for *B. subtilis* and 3 for *E. coli* under aerobic condition; however, realistic values are always lower and vary from case to case in different strains (Özkan et al. 2005; Sauer and Bailey 1999), precise determination of P/O ratio for a specific engineered strain requires labor-intensive measurements. The second challenge is the maintenance energy, which was found to be not only including the heat generation, but also related with cell mobility, protein and DNA turnover, osmoregulation, and pathway shift (Hoehler and Jorgensen 2013a). Traditionally, maintenance energy can be determined through employing chemostat culture at different dilution rates (Pirt 1965), which is not considered as a good reflection of maintenance energy now (Hoehler and Jorgensen 2013a). Due to the intricate and dynamic essence of maintenance energy and various types of genetic modifications, setting an appropriate value of maintenance energy for FBA also requires experimental assistant.

In order to address problems mentioned above, we propose an integrated platform of \(^{13}\)C-MFA and FBA in this work as isotopomer constrained flux balance analysis (icFBA, shown in
This method is utilized to explore the physiological status of several engineered E. coli strains that produce isobutanol (Baez et al. 2011a). Isobutanol, a promising biofuel potentially to replace ethanol, has attracted enormous attentions from both biochemical engineers and the public since 2008 (Atsumi et al. 2008a). However, there are still obstacles on the way to scale-up laboratory high yield isobutanol producing strains into industrial production. Several approaches have been attempted to work out the concerns such as product toxicity (Atsumi et al. 2010). A thorough examination on isobutanol producing strains by this fluxomics platform might help us to gain more insights into their physiology and provide some hints for industrial-scale fermentation. Furthermore, the knowledge obtained from this study can potentially applied in the design of other engineered strains.

We performed $^{13}$C labeling experiments on several isobutanol engineered E. coli strains. The growth characteristics and amino acid labeling patterns were used for fluxomics studies. Consequently, we obtained the flux distribution of central carbon metabolism. Subsequently, flux values of central metabolic pathways determined by $^{13}$C-MFA are taken as constraints for genome scale FBA calculation (iJO1366 is employed here) (Orth et al. 2011). Different values of maintenance energy, oxygen flux, as well as P/O ratio are also employed in the simulation of FBA.

### 2.3. Materials and Methods

#### 2.3.1. Strains and plasmids

BW25113 (CGSC# 7636) was the wild-type strain used in this study (Atsumi et al. 2008b) and it was purchased from the Escherichia coli Genetic Stock Center (http://cgsc.biology.yale.edu/). The knockout strain JCL260 ($\Delta$adhE, $\Delta$ldhA, $\Delta$frdBC, $\Delta$fnr, $\Delta$pta, $\Delta$pflB) and the plasmids pSA65 and pSA69 (Baez et al. 2011b; Atsumi et al. 2008c) were provided by Prof. James Liao
(University of California, Los Angeles). The plasmid pSA65 contains the genes \textit{kivd} and \textit{adhA} from \textit{Lactococcus lactis}. The plasmid pSA69 contains the genes \textit{alsS} of \textit{Bacillus subtilis} and \textit{ilvCD} of \textit{E. coli}. The detailed information of strains and plasmids are also listed in Table 2.1.

\textbf{2.3.2. Medium and culture conditions}

A minimal medium containing 0.5\% \([1, 2-^{13}\text{C}]\) glucose, M9 salts (Difco), and 10 mg/L vitamin B1 was used for the labeling experiments. The antibiotics, Ampicillin (100 μg/mL) and Kanamycin (50 μg/mL) were added as appropriate. For the pre-culture, a single colony of cells from a fresh plate was used to inoculate a 5 mL Luria-Bertani (LB) media. The pre-culture was grown overnight at 37°C on a rotary shaker at 225 rpm. The pre-culture was used to inoculate (0.2\%, v/v) a main culture of 10 mL minimal medium, grown on a rotary shaker in 250 mL shake flasks at 30°C and 225 rpm. The cultures were grown in duplicates and kept airtight by closing the flasks with rubber stoppers to prevent any loss of isobutanol due to evaporation. IPTG was added to a final concentration of 0.1 mM for induction. Samples were collected before and after (both at mid-log phase and late-log phase) the addition of IPTG. The liquid cultures (~10 mL) from each flask were centrifuged and the supernatant was separated from the biomass. Both the biomass and the supernatant samples were stored in -20°C prior to analysis.

\textbf{2.3.3. Quantification of biomass and extracellular metabolites}

Growth of the \textit{E. coli} cells was monitored by measuring the optical density of the cultures at 600 nm (OD\textsubscript{600}) using an Agilent Cary 60 UV-Vis spectrophotometer. The dry biomass concentration in gram per liter was determined based on the correlation of, OD\textsubscript{600} - 1 is equivalent to 0.338 g dry weight/L (Xiao \textit{et al.} 2013). The concentration of acetic acid and glucose were measured by using the corresponding enzymatic kit (R-Biopharm) and by following the manufacturer’s protocol. The enzymatic reactions were conducted at room temperature in a 96-well plate reader.
Isobutanol analysis was done using a gas chromatograph (GC) (Hewlett Packard model 7890A [Agilent Technologies] equipped with a DB5-MS column [J&W Scientific]) and a mass spectrometer (MS) (5975C, Agilent Technologies) (Xiao et al. 2012). Isobutanol was extracted from 800 µL of the supernatant using 400 µL of toluene as the extractant. Both the supernatant and the toluene were vortexed together for 2 minutes, followed by centrifugation at 10,000 × g for 5 min to separate the aqueous and organic phase. 1 µL of the organic layer was injected into the GC column with helium as the carrier gas. The GC oven was held at 70°C for 2 minutes and then raised to 200°C with a temperature ramp of 30°C min⁻¹, and the post run was set at 300°C for 6 minutes. The MS scan mode was set from m/z of 20 to m/z of 200. The quantification of isobutanol was done based on a calibration curve prepared with known concentrations of isobutanol ranging from 25 mg/L to 200 mg/L. Methanol was used as an internal standard for all the samples.

2.3.4. Mass isotopomer distribution of proteinogenic amino acids

The mass isotopomer distribution (MID) of proteinogenic amino acids were performed as described elsewhere (You et al. 2012). In short, the biomass pellets were hydrolyzed with 6 M hydrochloric acid and dried under a stream of air. The hydrolysates were dissolved in tetrahydrofuran and derivatized with N-Methyl-N-[tert-butylidimethyl-silyl] trifluoroacetamide (Sigma-Aldrich, MO) at 70°C for 1 hour. The amino acid analysis was also performed on the GC-MS equipped with a DB5-MS column. The sample injection volume was 1 µL and a 1:10 split ratio was utilized with helium as the carrier gas (1.2 mL/min). The GC oven was held at 150°C for 2 minutes and then raised to 280°C with a temperature ramp of 3°C min⁻¹, followed by another temperature ramp of 20°C min⁻¹ to a final temperature of 300°C and was held at 300°C for 5 minutes. The mass spectra were acquired from the MS with an m/z range of 60 to
The mass isotopomer distributions of the amino acids were corrected for the presence of naturally abundant isotopes \([^{13}\text{C}} - 1.13\%, \quad ^{18}\text{O} - 0.20\%, \quad ^{29}\text{Si} - 4.70\%, \quad \text{and} \quad ^{30}\text{Si} - 3.09\%\] using published algorithm (Tang et al. 2009a; Wahl et al. 2004).

**2.3.5. Central metabolic flux determined by \(^{13}\text{C}-\text{MFA}\)**

\(^{13}\text{C}-\text{MFA}\) was carried out based on the MID information from the proteinogenic amino acids. The metabolic network of \(E. \text{coli}\) strains includes major pathways such as the glycolysis, the pentose phosphate pathway, the Entner–Doudoroff (ED) pathway, the tricarboxylic acid (TCA) cycle, the glyoxylate shunt, and the anaplerotic pathways. For biomass flux, we adopted the same equation previously reported (He et al. 2014). In engineered strains that produce isobutanol, a simplified reaction was employed to describe the isobutanol flux \((2*\text{PYR} + \text{NADH} + \text{NADPH} \rightarrow \text{IB} + 2*\text{CO}_2)\). The detailed information of metabolic network is listed in Table 2.3. The carbon substrate uptake rate was defined as 100, while other metabolic fluxes were normalized to a scale of 100. The energy metabolism was not included in \(^{13}\text{C}-\text{MFA}\) calculation, because different P/O ratio values affect the central carbon flux profiles seriously (data not shown). The EMU (elementary metabolite units) method was adopted for \(^{13}\text{C}-\text{MFA}\). The MATLAB-based software WUFlux, developed by Tang lab, was employed for \(^{13}\text{C}-\text{MFA}\) calculation (available in \(13\text{cmfa.org}\)). 100 randomly generated initial values were used and the solution set with minimal objective function value (best fit) was selected as the final fluxes. The relative flux profile in each strain was calculated by minimizing the difference between predicted and measured isotopomer patterns. The measured production rates of acetic acid and biomass are used as the constraints for \(^{13}\text{C}-\text{MFA}\), while isobutanol flux is predicted by the flux model. Since the precise measurement of volatile extracellular metabolites (i.e., acetate) is difficult, calculation of \(^{13}\text{C}-\text{MFA}\) did not place tight constraints on overflow metabolite fluxes. Due to its high volatility,
isobutanol is difficult to be determined precisely thus the measured value will only be used as reference.

2.3.6. Genome-scale model constrained by $^{13}$C-MFA flux

The *E. coli* iJO1366 genome scale model was employed for FBA studies was minor modifications (Orth et al. 2011): the isobutanol flux ($3\text{mob}[c] + \text{nadh}[c] + h[c] \rightarrow \text{isobutanol} + \text{nad}[c] + \text{co2}[c]$) was added for engineered strains; a few specific fluxes were knockout for JCL260 strain and BW25113 strain (Monk et al. 2013). The $^{13}$C-MFA flux $v^\text{mfa}$ was used as the boundaries to constrain central metabolic flux of genome scale model using the following rules (Blank et al. 2005):

$$s.t. \quad S \cdot v' = 0$$
$$v^\text{mfa}_i \cdot (1 - \delta) \leq v'_i \leq v^\text{mfa}_i \cdot (1 + \delta)$$
$$\delta = 20\%$$

The objective function is to maximize isobutanol production (for engineered strains, set growth rates to be experimental values) or to maximize growth rate (for control strain). Default boundary conditions of iJO1366 were employed here, except for the fluxes of glucose uptake, oxygen uptake, and maintenance energy. We test the sensitivities of oxygen uptake rate, the maintenance energy, and the P/O ratio on the isobutanol production flux. Considering of oxygen supply conditions in our $^{13}$C-MFA experiments, we set the oxygen flux to be 16 mmol/gCDW·h (Varma et al. 1993). For isobutanol production by strain 5 (JCL260/pSA65+69, see Table 2.1.) in the bioreactor, we employed a modified biomass equation based on the $^{13}$C labeling information -- non-labeled biomass derived from yeast extract was deducted (Xiao et al. 2012), as well as a reduced oxygen flux (average 13 mmol/gCDW·h) (Xu et al. 1999).
To access strain stability, we also employ evolutionary fitness to quantify each strain: measured growth rate relative to the predicted maximal growth rate by FBA under the same conditions (Steinmetz et al. 2002).

2.4. Results and discussions

2.4.1. Physiological states of different Strains

We performed labeling experiments by feeding *E. coli* strains with 1, 2-$^{13}$C labeled glucose in minimal medium under aerobic conditions. The physiological states of different strains are recorded in Table 2.2.: As expected, the control strain (JCL260, strain 1) has a much higher specific growth rate compared with the other three isobutanol-producing (strains 2 – 4). Significant lower biomass yield was observed in engineered strains compared with control strain. JCL260 has higher efficiency for biomass yield while JCL260/pSA65 displays the lowest biomass yield compared with BW25113 engineered strain. Acetic acid production was found in cases of two BW25113 host strains, but was not detected for either JCL260 strain in the minimal medium. Strain JCL260/(pSA65+69) demonstrated poor or no growth repeatedly in minimal medium even without IPTG induction. Thereby, we used previous experimental data of JCL260/(pSA65+69) growth in presence of yeast extract, which was published by our lab three years ago (strain 5, in Table 2.2.) (Xiao et al. 2012). Apparently, strain 5 exhibited a significantly higher yield of isobutanol in the presence of yeast extract, compared with other strains growth in a minimal medium. Previous reports have proved that yeast extract mainly contributed to the synthesis of biomass building blocks while most of isobutanol was converted from glucose (Xiao et al. 2012). However, yeast extract is indispensable for this process: Isobutanol production pathway poses considerable metabolic burdens on host strains, thereby, cellular energy generated from glucose catabolism is insufficient to power both biomass
synthesis and isobutanol production in high-yield strain JCL260/(pSA65+69). Biomass synthesis, especially the biosynthesis of amino acids consumes lots of ATP and NADH (Russell and Cook 1995; Akashi and Gojobori 2002), while yeast extract relieves such burden by supplying most amino acids (Xiao et al. 2012; Selvarasu et al. 2009). Notably, isobutanol itself will disrupt cell membrane integrity and reduce the efficiency of oxidative phosphorylation, leading to decreased pH and ATP supply (Atsumi et al. 2010; Wu et al. 2015). Supplement of yeast extract relieves the burden from energy intensive biomass synthesis, thereby well solves the energy crisis inside isobutanol producing strains.

2.4.2. Central metabolic flux determined by \(^{13}\text{C}\)-MFA

To quantify the intracellular fluxes of the central metabolic pathway, \(^{13}\text{C}\)-MFA was employed to solve the flux profile based on the mass isotopic distribution (MID) of proteinogenic amino acid. The results of the central flux map are illustrated in Figure 2.2a-d. Central metabolic flux profile always undergoes apparent adjustments in response to metabolic burdens and cell stress within engineered strains (He et al. 2014; Antoniewicz et al. 2007c). In this work, significant changes in central metabolism were observed in response to the genetic alternations and isobutanol production for engineered strain 2 (JCL260/pSA65): The pentose phosphate (PP) pathway is up-regulated in engineered strains to balance NADPH demanding (compared with JCL260, strain 1); the glyoxylate shunt increases to reduce the carbon loss as CO\(_2\); while the anaplerotic flux reduced significantly, enable more flux from glucose rerouting into pyruvate that is a precursor for isobutanol synthesis. The glucose-inhibited glyoxylate shunt is recovered by a decreased glucose uptake flux (inhibited by isobutanol) (Atsumi et al. 2010; Bowles and Ellefson 1985), while knocking out \(ppc\) gene leads to increase activity of glyoxylate shunt (Fong et al. 2006).
Compared with BW25113 strains (strain 3 and 4), JCL260 strains (1, 2) only produced little amount of acetate in the minimal medium, which was even out of detection limit (Table 2.2). Knocking out acetate related genes not only leads to the loss of acetate production capability in JCL260, but also reroutes more carbon fluxes into isobutanol production pathway. However, significant acetate production was still observed for strain JCL260/(pSA65+69) in the presence of yeast extract. A possible source of acetate is the biosynthesis and degradation of amino acids (e.g., $\text{SER} + \text{AceCoA} + 3 \text{ ATP} + 4 \text{ NADPH} \rightarrow \text{CYS} + \text{Ac}$), especially supplied with abundant exogenous amino acids (i.e., yeast extract).

2.4.3. Energy metabolism and evolutionary fitness analysis

To illustrate the energy status of different strains under genetic modifications (e.g., gene knockout, exogenous plasmids) and internal stress (e.g., protein overexpression, isobutanol toxicity), we performed a simple analysis by assuming ideal energy metabolism (i.e., P/O ratio = 3) and the results are shown in Table 2.4 and Figure 2.3. In ideal condition (Figure 2.3a), the control strain (JCL260) has highest excess energy while strain 2 (JCL260/pSA65) is of lowest excess energy; this situation overturned when inefficient energy metabolisms (P/O ratio = 1, shown in Figure 2.3b) apply for energy analysis. P/O ratio of 1 is closer to the real situations of $^{13}$C-MFA experiments, based on literature reports on *E. coli* strains (Özkan *et al.* 2005; Noguchi *et al.* 2004). The ‘excess energy’ actually is used as cellular maintenance cost, whereas the control strain (strain 1) has the lowest maintenance energy, while strain 2 (JCL260/pSA65) shows the highest maintenance energy cost. Further, strain 4 (BW/pSA65+69) has a higher maintenance requirement compared with strain 3 (BW/pSA65). Such differences in maintenance cost are caused by additional energy expense used for extra plasmid maintenance, as well as heterogeneous protein expression (Glick 1995).
Strain stability is a crucial factor for industrial-scale fermentation. Evolutionary fitness is used to estimate the genetic stability of engineered strains, based on the assumption that cells with fastest growth at given conditions has the largest fitness of survival (Blank et al. 2005; Steinmetz et al. 2002). Considering of complicated genetic modifications in this study, evolutionary fitness is used here rather than physiological fitness (Blank et al. 2005). From the results shown in Figure 2.4, strain 1 (JCL260) has the highest evolutionary fitness while strain 4 (BW/pSA65+69) demonstrates the lowest degree of evolutionary fitness. Besides, strain 5 may have even lower evolutionary fitness in the minimal medium because it contains dual plasmids in addition to a series of gene knockouts. It is well known that maintenance of exogenous plasmids brings considerable metabolic burdens that lead to increased instability of genotype (Silva et al. 2012). Researchers have proposed a series of approaches to reduce the negative impacts from plasmids, such as employment of low copy plasmid (Jones et al. 2000) or insertion of genes into chromosome (Tyo et al. 2009). With advancements of novel genome editing tools (e.g., CRISPR, TALEN), genetic modifications at chromosome level will result in strains with improved genetic stability in the future (Luo et al. 2015).

2.4.4. The effects of P/O ratio, oxygen flux, and maintenance energy on isobutanol production

One major bottleneck for metabolic engineering is that laboratory high-yield strains fail to make good performance during the scale-up process. Quantification of several factors involving this process (e.g., environmental factors, such as oxygen, nutrient availability, and strain physiological factors such as P/O ratio, maintenance energy) may provide novel insights into this problem. In this work, we employed FBA to investigate the effects of oxygen flux, nutrient availability, and maintenance energy on product yield. In particular, central metabolic fluxes determined by $^{13}$C-MFA are utilized to constrain flux boundary of FBA to ensure the final flux
profile to be in a reasonable range (Blank et al. 2005). Meanwhile, we also determine the effect of yeast extract on isobutanol production based on the $^{13}$C labeling data of strain 5. The details of simulations are described in section 2.3.5, and the results are presented in Figure 2.5 and 2.6.

The maximal yield of isobutanol differs from strain to strain: strain 5 > strain 2 > strain 4 > strain 3. This prediction exactly matches the experimental observations of isobutanol yield recorded in Table 2.2. Meanwhile, isobutanol production is more sensitive to the variation in maintenance energy and oxygen flux, rather than P/O ratio. Isobutanol synthesis requires NADH/NADPH rather than ATP in its synthetic pathway, which explains its robustness to P/O ratio change (Wu et al. 2015). Oxygen is not required for isobutanol synthesis; however, oxygen-involved oxidative phosphorylation contributes most energy for biomass synthesis at growth phase. The competition between biomass growth/protein expression and isobutanol production on intracellular energy makes ATP-rich oxidative respiration pathway very favorable, leading to increased sensitivity to oxygen concentration. During industrial-scale fermentations, environmental conditions (e.g., pH, oxygen concentration) at different locations of the bioreactor are changing with the time (Garcia-Ochoa and Gomez 2009; Zou et al. 2012). Further, massive consumption of oxygen and insufficient mixing during exponential growth leads to heterogeneous oxygen distribution within the reactor. The conflict between increasing oxygen demand for cell growth and decreasing oxygen supply capability in many regions of bioreactor always leads to poor performance of engineered strains (fall off the cliff of isobutanol production as shown in Figure 2.5 and 2.6). In many cases, supply of rich nutrients will not only lesson the cost of carbon source, and energy/reducing power (ATP, NADH, and NADPH) on amino acids synthesis, but also alleviate the requirement on activation of corresponding synthesis pathways, that’s why strain 5 have the best performance of all test conditions in this study (results shown in
Table 2.2, Figure 2.5e, and Figure 2.6e). In practice, induction of protein overexpression at mid or late growth phase also can relieve the resource crisis mentioned above (Xu et al. 2014). Alcohol production process has an apparent advantage: Even cell membrane and oxidative phosphorylation system are destructed by aldehyde and alcohol, the synthesis process of isobutanol is still going on.

2.5. Conclusions

In this work, we have resolved the central carbon metabolism of isobutanol-producing *E. coli* strains by $^{13}$C-MFA. The results indicated that genetically modified strains can make adjustments over their flux profile in response to the requirements of isobutanol production. Also, extensive genetic modifications will lead to decreased evolutionary fitness as well as increased maintenance cost. Further, isobutanol production is very sensitive to the increase of cellular maintenance energy while rich nutrients (e.g., yeast extract) can relieve the stress caused by metabolic burdens. Lastly, genome editing will bring less metabolic burden and more evolutionary fitness compared with plasmid-based modification, thus is suitable for bioprocess scale-up in the future.

2.6. References


<table>
<thead>
<tr>
<th>Strain/ Plasmid</th>
<th>Genetic Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>JCL260 (Strain)</td>
<td>BW25113/F[traD36, proAB+, lacIq ZΔM15] ΔadhE, ΔfrdB, Δfnr, ΔldhA, Δpta, ΔpflB</td>
</tr>
<tr>
<td>pSA65 (Plasmid)</td>
<td>ColElori;AmpR; PLlacO1: kivd-adhA (Lactococcus lactis)</td>
</tr>
<tr>
<td>pSA69 (Plasmid)</td>
<td>P15ori;KanR; PLlacO1: alsS(Bacillus subtilis)-ilvCD</td>
</tr>
</tbody>
</table>

Table 2.1 Detailed information of plasmids and strains used in this study

<table>
<thead>
<tr>
<th>NO</th>
<th>Host Strain/ Plasmid</th>
<th>Growth rate (h⁻¹)</th>
<th>Biomass yield (g/g glucose)</th>
<th>Acetate yield (g/g glucose)</th>
<th>Isobutanol yield (g/g glucose)</th>
<th>Glucose uptake rate (mmol/gDW*h)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>JCL260</td>
<td>0.345</td>
<td>0.432</td>
<td>~0</td>
<td>0</td>
<td>3.921</td>
</tr>
<tr>
<td>2</td>
<td>JCL260/pSA65</td>
<td>0.126</td>
<td>0.240</td>
<td>~0</td>
<td>0.143</td>
<td>3.968</td>
</tr>
<tr>
<td>3</td>
<td>BW25113/pSA65</td>
<td>0.095</td>
<td>0.353</td>
<td>0.065</td>
<td>0.098</td>
<td>1.943</td>
</tr>
<tr>
<td>4</td>
<td>BW25113/(pSA65+pSA69)</td>
<td>0.091</td>
<td>0.354</td>
<td>0.114</td>
<td>0.097</td>
<td>3.249</td>
</tr>
<tr>
<td>5</td>
<td>JCL260/(pSA65+pSA69)*</td>
<td>0.112</td>
<td>0.390</td>
<td>0.350</td>
<td>0.360</td>
<td>7.589</td>
</tr>
</tbody>
</table>

* Data adopted from previous experiments published by Tang lab, with 5 g/L yeast extract in bioreactor (Xiao et al. 2012).

Physiological information of strain 1-4 came from ¹³C-MFA experiments in this study, while physiological data of strain 5 was from a paper published by our lab.

Table 2.2 Physiological information of strains used in this study
<table>
<thead>
<tr>
<th>Reaction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Glucose + ATP == G6P</td>
</tr>
<tr>
<td>G6P == F6P</td>
</tr>
<tr>
<td>F6P + ATP == FBP</td>
</tr>
<tr>
<td>FBP == F6P</td>
</tr>
<tr>
<td>FBP == DHAP + GAP</td>
</tr>
<tr>
<td>DHAP == GAP</td>
</tr>
<tr>
<td>GAP == G3P + ATP + NADH</td>
</tr>
<tr>
<td>G3P == PEP</td>
</tr>
<tr>
<td>PEP == PYR + ATP</td>
</tr>
<tr>
<td>PYR + 2*ATP == PEP</td>
</tr>
<tr>
<td>PYR == AceCoA + CO2 + NADH</td>
</tr>
<tr>
<td>AceCoA + OAA == CIT</td>
</tr>
<tr>
<td>CIT == ICIT</td>
</tr>
<tr>
<td>ICIT == AKG + CO2 + NADPH</td>
</tr>
<tr>
<td>AKG == SucCoA + CO2 + NADH</td>
</tr>
<tr>
<td>SucCoA == SUC + ATP</td>
</tr>
<tr>
<td>SUC == FUM + FADH2</td>
</tr>
<tr>
<td>FUM == MAL</td>
</tr>
<tr>
<td>MAL == OAA + NADH</td>
</tr>
<tr>
<td>MAL == PYR + CO2 + NADH</td>
</tr>
<tr>
<td>MAL == PYR + CO2 + NADPH</td>
</tr>
<tr>
<td>PEP + CO2 == OAA</td>
</tr>
<tr>
<td>OAA + ATP == PEP + CO2</td>
</tr>
<tr>
<td>ICIT == GLX + SUC</td>
</tr>
<tr>
<td>GLX + AceCoA == MAL</td>
</tr>
<tr>
<td>G6P == PG6 + NADPH</td>
</tr>
<tr>
<td>PG6 == CO2 + Ru5P + NADPH</td>
</tr>
<tr>
<td>Ru5P == X5P</td>
</tr>
<tr>
<td>X5P == R5P</td>
</tr>
<tr>
<td>X5P + R5P == GAP + S7P</td>
</tr>
<tr>
<td>GAP + S7P == E4P + F6P</td>
</tr>
<tr>
<td>X5P + E4P == GAP + F6P</td>
</tr>
<tr>
<td>PG6 == PYR + GAP</td>
</tr>
<tr>
<td>AceCoA == Ac + ATP</td>
</tr>
<tr>
<td>AKG + NADPH == GLU</td>
</tr>
<tr>
<td>GLU + ATP == GLN</td>
</tr>
<tr>
<td>GLU + ATP + 2*NADPH == PRO</td>
</tr>
<tr>
<td>GLU + GLN + CO2 + ASP + AceCoA + 5*ATP + NADPH == ARG + AKG + FUM + Ac'</td>
</tr>
<tr>
<td>OAA + GLU == ASP + AKG</td>
</tr>
<tr>
<td>ASP + 2*ATP == ASN</td>
</tr>
<tr>
<td>PYR + GLU == ALA + AKG</td>
</tr>
<tr>
<td>G3P + GLU == SER + AKG + NADH</td>
</tr>
<tr>
<td>Chemical Reaction</td>
</tr>
<tr>
<td>-------------------</td>
</tr>
<tr>
<td>'SER == GLY + Methylene_THF'</td>
</tr>
<tr>
<td>'GLY == Methylene_THF + CO2 + NADH'</td>
</tr>
<tr>
<td>'Methylene_THF + NADH == Methyl_THF'</td>
</tr>
<tr>
<td>'Methylene_THF == Formyl_THF + NADPH'</td>
</tr>
<tr>
<td>'ASP + 2<em>ATP + 2</em>NADPH == THR'</td>
</tr>
<tr>
<td>'THR == GLY + AceCoA + NADH'</td>
</tr>
<tr>
<td>'SER + AceCoA + 3<em>ATP + 4</em>NADPH == CYS + Ac'</td>
</tr>
<tr>
<td>'ASP + PYR + GLU + SucCoA + ATP + 2*NADPH == LYS + CO2 + AKG + Suc'</td>
</tr>
<tr>
<td>'ASP + Methyl_THF + CYS + SucCoA + ATP + 2*NADPH == MET + PYR + Suc'</td>
</tr>
<tr>
<td>'GLU + NADPH + 2*PYR == VAL + AKG + CO2'</td>
</tr>
<tr>
<td>'AceCoA + 2<em>PYR + GLU + NADPH == LEU + AKG + NADH + 2</em>CO2'</td>
</tr>
<tr>
<td>'THR + PYR + GLU + NADPH == ILE + AKG + CO2'</td>
</tr>
<tr>
<td>'E4P + 2*PEP + GLU + ATP + NADPH == PHE + AKG + CO2'</td>
</tr>
<tr>
<td>'E4P + 2*PEP + GLU + ATP + NADPH == TYR + AKG + NADH + CO2'</td>
</tr>
<tr>
<td>'SER + R5P + 2<em>PEP + E4P + GLN + 3</em>ATP + NADPH == TRP + GAP + PYR + GLU + CO2'</td>
</tr>
<tr>
<td>'R5P + Formyl_THF + GLN + ASP + 5<em>ATP == HIS + AKG + FUM + 2</em>NADH'</td>
</tr>
<tr>
<td>'NADH == NADPH'</td>
</tr>
<tr>
<td>'NADH == 3*ATP'</td>
</tr>
<tr>
<td>'FADH2 == 2*ATP'</td>
</tr>
<tr>
<td>'Ac == 0*Ex'</td>
</tr>
<tr>
<td>'CO2 == 0*Ex'</td>
</tr>
<tr>
<td>'CO2_air + CO2 == CO2_cell + CO2'</td>
</tr>
</tbody>
</table>

Table 2.3 Metabolic network for $^{13}$C-MFA calculation
<table>
<thead>
<tr>
<th></th>
<th>NADH</th>
<th>NADPH</th>
<th>ATP</th>
<th>FADH₂</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Glycolysis</td>
<td>169</td>
<td>175</td>
<td>173</td>
<td>175</td>
</tr>
<tr>
<td>PP pathway</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TCA cycle</td>
<td>214</td>
<td>119</td>
<td>172</td>
<td>162</td>
</tr>
<tr>
<td>Amino acid synthesis</td>
<td>25</td>
<td>9</td>
<td>16</td>
<td>12</td>
</tr>
<tr>
<td>Acetic acid formation</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Biomass formation</td>
<td>13</td>
<td>7</td>
<td>10</td>
<td>9</td>
</tr>
<tr>
<td>Isobutanol production</td>
<td>0</td>
<td>-43</td>
<td>-20</td>
<td>-20</td>
</tr>
<tr>
<td>One-carbon metabolism</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>Anaplerotic Pathway</td>
<td>7</td>
<td>15</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>carbon metabolism</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

All flux values are normalized to a glucose uptake rate of 100 mol/h in each strain.

a. The anaplerotic pathway is assumed to only produce NADH and consume minimal amount of ATP.

b. The excessive NADH & FADH₂ are assumed to be converted to ATP via oxidative phosphorylation at maximal P/O ratio (NADH \( \rightarrow \) 3 ATP, FADH₂ \( \rightarrow \) 2 ATP)

Table 2.4 Energy metabolism of different strains
Figure 2.1 Diagram of $^{13}$C flux as constraints for FBA
1. JCL260
2. JCL260/pSA65
3. BW25113/pSA65;
4. BW25113/(pSA65 + pSA69).

Figure 2.2. Central metabolic flux determined by $^{13}$C-MFA on control strain (1. JCL260) and three engineered strains (2. JCL260/pSA65; 3. BW25113/pSA65; 4. BW25113/(pSA65 + pSA69)).
Figure 2.3a Energy analysis of four strains at ideal energy condition (P/O ratio = 3)

Figure 2.3b Energy analysis of four strains with low energy metabolism (P/O ratio = 1)
Figure 2.4 Evolutionary fitness of four strains in this study

2.5a. Influence of P/O ratio and maintenance energy on growth rate of strain 1
2.5b. Influence of P/O ratio and maintenance energy on isobutanol production of strain 2

2.5c. Influence of P/O ratio and maintenance energy on isobutanol production of strain 3
2.5d. Influence of P/O ratio and maintenance energy on isobutanol production of strain 4

2.5e. Influence of P/O ratio and maintenance energy on isobutanol production of strain 5

Figure 2.5a-e: Influence of P/O ratio and maintenance energy on isobutanol production potential (growth rate for strain 1, JCL260), simulated by FBA
2.6a. Influence of oxygen uptake flux and maintenance energy on growth rate of strain 1

2.6b. Influence of oxygen uptake flux and maintenance energy on isobutanol production of strain 2
2.6c. Influence of oxygen uptake flux and maintenance energy on isobutanol production of strain 3

2.6d. Influence of oxygen uptake flux and maintenance energy on isobutanol production of strain 4
2.6e. Influence of oxygen uptake flux and maintenance energy on isobutanol production of strain
Figure 2.6.a-e: Influence of oxygen uptake flux and maintenance energy on isobutanol production potential (growth rate for strain 1, JCL260), simulated by FBA. Default P/O ratio of 1.75 is employed here.
CHAPTER THREE

INVESTIGATE ENERGY METABOLISM OF MICROBIAL CELL FACTORIES BY YIN-YANG THEORY

3.1. Abstract

In ancient Chinese philosophy, Yin-Yang describes two contrary forces that are interconnected and interdependent. This concept also holds true in microbial cell factories, where Yin represents energy metabolism in the form of ATP, and Yang represents carbon metabolism. Current biotechnology can effectively edit the microbial genome or introduce novel enzymes to redirect carbon fluxes. However, the limitation of the internal powerhouse prevents cells from achieving high carbon yields and rates. It is because that microbial metabolism could lose over 60% of free energy as heat when converting sugar into ATP; while high cell maintenance in microbial hosts further aggravates cellular ATP shortage. Via a flux balance analysis model, we further demonstrate the penalty of ATP expenditure on biofuel synthesis. To ensure cell powerhouse being sufficient for microbial cell factories, we propose five principles: 1. Take advantage of native pathways for product synthesis. 2. Pursue biosynthesis relying only on pathways or genetic parts without significant ATP burden. 3. Combine microbial production with chemical conversions (semi-biosynthesis) to reduce biosynthesis steps. 4. Create ‘minimal cells’ or use non-model microbial hosts with higher energy fitness. 5. Develop a photosynthesis chassis that can utilize light energy and cheap carbon feedstocks. Meanwhile, metabolic flux analysis can be used to quantify both carbon and energy metabolisms and determine ‘the straw that broke
the camel’s back’. The fluxomics results are essential to evaluate the industrial potential of laboratory strains, avoiding false starts and dead ends during metabolic engineering.

**Key words:** ATP, energy metabolism, flux analysis, free energy, maintenance loss, semi-biosynthesis

### 3.2. Introduction

In the past decade, molecular biology tools have developed rapidly and now offer new opportunities for metabolic engineering of microbial hosts (Sun and Zhao 2013; Jiang *et al.* 2013; Pratt and MacRae 2009; Qi *et al.* 2013; Wang *et al.* 2009; Isaacs *et al.* 2011). These tools include the selection of plasmids with different copy numbers, promoter engineering, codon optimization, synthetic scaffolds, directed evolution or rational design of enzymes, ribosome binding sites editing, and competitive pathways deletion. Advanced genome engineering (e.g., CRISPRs and TALENs) and automation of conventional genetic techniques (e.g., MAGE) provide efficient capabilities for editing genomes and evolving new functions. At the same time, systems biology (e.g., genomics, transcriptomics, and proteomics) can characterize complex cell networks, mine useful genes, discover new enzymes, reveal metabolic regulations, and screen mutant phenotypes. The advent of these powerful tools seems to lead researchers into a new epoch of bioprocess industries using GMOs (genetically modified organisms) in the near future. However, that is not the whole story.

The golden age of industrial biotechnology dawned in the early 1940s, driven by the mass production of penicillin and enjoyed a fast growth in the 1950s~1980s. Microbial bioprocess has produced diverse commodity chemicals (such as ethanol, amino acids, citric acid,
and lactate) as well as recombinant proteins and antibiotics in the last century. Those commercial products mainly rely on natural strains or strains with minor genetic modifications (usually only one or few new genes). Since the recent decade, in the hope of producing chemicals at low costs and reducing greenhouse gas emissions, an enormous amount of investment has been devoted to metabolic engineering in many nations. Although modern biotechnologies can engineer cells to synthesize diverse products in laboratories, there are only a few GMO products that have become commercially promising in the past decade (e.g., artemisinic acid and 1, 4-butanediol). GMOs are particularly used for biofuel manufacturers, such as short-chain alcohols, fatty acid derived chemicals, and isoprenoid-based biofuels (Atsumi et al. 2009; Lindberg et al. 2010; Oliver et al. 2013). For example, Gevo and Butamax introduce the keto-acid/Ehrlich pathway into yeasts to produce isobutanol (Nielsen et al. 2014). Amyris extend the mevalonate pathway in Saccharomyces cerevisiae for branched and cyclic terpenes (e.g., farnesene) synthesis. However, these biofuel processes have not achieved strong net profit margin yet. To date, the industrial-scale biofuel is still ethanol, which is cheaply manufactured from sugar cane in Brazil. In this perspective, we will address one of the hidden constraints in microbial cell factories (i.e., Energy metabolism).

3.3. The energy losses in microbial cell factories

Heterotrophic organisms obtain free energy in the form of ATP by breaking organic substrates into CO₂ (Figure 3.1). Theoretically, oxidation of one mole of glucose to CO₂ (ΔcH°298 ≈ -2.8 MJ/mol) can generate 38 moles of ATP. Hydrolysis of these ATP to ADP (ΔG° = -30.5 kJ/mol) provide ~1.2 MJ of biochemical energy. Thereby, ~60% of energy from glucose is lost as heat during ATP synthesis (similar to a Carnot heat engine). Cell consumes ATP for diverse
activities, such as nutrient/metabolite transport, chemotaxis, chemical gradient preservation, biomass component repair, and macromolecule re-synthesis (Hoehler and Jorgensen 2013b). These maintenance costs, essential for cell survival and stress adaptation, compete for ATP resources for biomass growth and product synthesis. On the other hand, microbial hosts have not evolved towards optimal energy metabolism. Over billions of years of evolution, microbes with a higher growth rate gained a selective advantage when competing for shared energy resources, but these fast growing species have a lower yield of ATP from substrates (e.g., less than 30 ATP/glucose) (Pfeiffer et al. 2001). The oxidative phosphorylation (P/O) ratio represents ATP generation efficiency through substrate oxidation. Theoretically, three ATP can be obtained from the reduction of one oxygen atom (i.e., P/O = 3) during oxidative phosphorylation. Although slow-growing mammalian cells can achieve P/O values close to 3, bacteria and yeasts often have P/O ratios below 2.5 (note: these microbes may dissipate the proton gradient before it can be fully used for charging the ATP synthase). Secondly, microbial hosts may lose ATP yield due to byproducts synthesis, membrane leakage, removal of reactive oxygen species, or suboptimal cultivations (insufficient mixing, shear stress, or biofilm formation). Lastly, the electron transport chain for ATP generation and nutrient transporters may compete for membrane and intracellular spaces so that the capacity of the microbial powerhouse cannot be easily upgraded (Ibarra et al. 2002; MacLean and Gudelj 2006).

We introduce the terminology “metabolic entropy” to define the free energy in the substrates that is lost through cellular energy metabolism and becomes unavailable for biosynthesis. Metabolic entropy has gained attention from metabolic flux analysis researchers because the objective function of biomass production in FBA (flux balance analysis) always overestimates microbial growth rates. Moreover, FBA predictions highly depend on the
assumption of a fixed ATP maintenance coefficient. To address this problem, researchers developed $^{13}$C-metabolic flux analysis (MFA) to quantify the microbial “metabolic entropy” directly via tracer experiments. By examining *Bacillus subtilis* mutants, $^{13}$C-MFA has discovered that the suboptimal cell metabolism is associated with the increased energy usage in the face of environmental and random genetic perturbations (Fischer and Sauer 2005). This study suggests that mutating regulatory genes can drive carbon fluxes towards the desired pathways; however, such mutations reduce ATP availability for adaptive responses under adverse environmental conditions (i.e., metabolic engineering achieves microbial productivity by sacrificing their energy fitness).

3.4. The tradeoff between product yield and energy fitness

Traditional metabolic engineering uses plasmids and heterologous enzymes to redirect carbon fluxes. Early studies have shown high copy number plasmids cause significant alterations in cell properties and strongly influence metabolic engineering endeavors (Birnbaum and Bailey 1991). $^{13}$C-MFA of *E. coli* strains revealed higher acetate production and O$_2$ uptake rates in plasmid-containing strains than in the plasmid-free strains (Wang *et al.* 2006a). The presence of plasmids can increase cell maintenance, decrease growth rate and change intracellular fluxes, especially suppressing the oxidative pentose phosphate pathway (Ow *et al.* 2009). Similarly, synthetic biology parts (such as novel pathways, protein scaffolds, and genetic circuits) may also increase metabolic entropy if extra nucleic acids and proteins are required to be made by the hosts (note: elongation of one amino acid costs four ATP molecules) (Stephanopoulos *et al.* 1998a). In reality, microbial systems have frugal enzymatic machinery (each native enzyme in a single *E. coli* cell may only have dozens of molecule copies and places minimal biosynthesis burden on cell metabolism) (Taniguchi *et al.* 2010). During pathway engineering, optimizing
enzyme levels is difficult because a large portion of over-expressed enzymes may be inactivated due to protein misfolding. Considerable ATP expenditure for heterologous enzyme synthesis can trigger stress responses and alternate hosts’ physiology. For example, \(^{13}\text{C}\)-MFA has been used to examine metabolic burdens in \textit{E. coli} during biosynthesis of recombinant proteins. The results indicate a 25% increase in the total ATP expenditure rate in the highest yielding strain (up to 45 mmol ATP/g CDW/h) (Heyland \textit{et al.} 2011). To overcome such an energy limitation, \textit{E. coli} has to reduce biomass synthesis and enhance oxidative phosphorylation for ATP generation. Besides, microbial hosts often suffer from increased non-growth associated maintenance as well as reduced respiration efficiency (poor P/O ratio) due to membrane stresses (Sauer and Bailey 1999; Varma and Palsson 1994). Therefore, introduction of an extended heterologous pathway into a microbial host often causes deleterious effects on cell metabolism. Ultimately, the host will lose the capability to grow in a minimal carbohydrate medium; while rich nutrients, such as yeast extract (producing 1 g of yeast extract consumes 3 g of glucose), have to be supplied to relieve the cell’s energy burden (Xiao \textit{et al.} 2012).

Our theory of energy burden can guide strain development to tolerate product stresses. For instance, an isobutanol-tolerant mutant has been isolated after serial transfers. However, the final isobutanol productivity of this evolved strain did not show improvement (Atsumi \textit{et al.} 2010). The export systems (e.g., ABC transporters) have been engineered for recovering cell growth under biofuel stresses (Dunlop \textit{et al.} 2011). The ATP-driven efflux pumps show limited enhancement of short-chain alcohol productivity (\(~10\%\)) (Foo \textit{et al.} 2014). On the other hand, efflux pumps work well when they are introduced into low-performance strains, in which their product titers are well below 1 g/L (Dunlop \textit{et al.} 2011). These observations explain the fact that cell adaptation to a stressful environment may require ATP expenditure and thus induce
significant metabolic burdens on biosynthesis (Zhang and Lynd 2005). For the same reason, tolerance engineering often works well on yeast strains for ethanol production because of simple synthesis pathway and net ATP generation from glycolysis. For example, engineering transcriptional machinery or up-regulation of the potassium/proton pumps in *Saccharomyces cerevisiae* can significantly improve ethanol tolerance and the production titer (well above 100 g/L) (Alper *et al.* 2006; Lam *et al.* 2014). In conclusion, when the microbial hosts already have high biosynthesis burdens, we should focus on specific regulatory genes rather than efflux pumps. For example, a methionine biosynthesis regulator can significantly improve both biofuel tolerance and productivity in *Escherichia coli* (Foo *et al.* 2014). In yet another case, the inactivation of a histidine kinase may enhance the butanol productivity in *Clostridium acetobutylicum* by delaying cell sporulation (Xu *et al.* 2015).

### 3.5. Sensitivity analysis of the energy penalty on biofuel synthesis

We employ a genome-scale flux balance model (iJO1366) to simulate the adverse impacts of *E. coli* energy metabolism on biofuel product yields (Figure 3.2) (Orth *et al.* 2011). Apart from the intracellular stress caused by enzyme overexpression, the release of large amounts of biofuel molecules (alcohol or fatty acid) will interfere enzymatic reactions *in vivo* and disrupt the cellular membrane’s integrity, which results in reduced efficiencies of oxidative respiration (Lennen *et al.* 2011; Atsumi *et al.* 2010). Thereby, metabolic engineering approaches are effective in redirecting carbon fluxes to biosynthesis only in these low-productivity strains whose energy metabolism are not overloaded. We have FBA test the penalty of metabolic burdens (such as maintenance cost) and the decrease of P/O ratio on biofuel yields. The simulations show that microbial energy metabolism is usually abundant so that they can support
certain amount of metabolic burdens without having apparent biosynthesis deficiency (e.g., without having a slower growth). However, cell burden may increase during the routine genetic modifications. When cell powerhouse is unable to fully afford the increasing ATP expenditure, the biosynthesis yield will have a sudden drop (i.e., the straw that broke the camel’s back), forming a cliff in Figure 3.2.

FBA simulations yield two insights into microbial biofuels. First, alcohol (ethanol and isobutanol) producing *E. coli* strains not only have higher carbon yields (0.67 C-product/C-glucose), but also are insensitive to P/O ratios (Figure 3.2a, b). Comparing to isobutanol, ethanol production is less sensitive to the metabolic burden (larger energy sufficient zone). Anaerobic ethanol fermentation, an ancient bioprocess from the beverage industry, does not need additional energy from O₂, lowering its process costs. From a stoichiometric perspective, glycolysis generates two net ATP per glucose, which fulfills the cell energy expenditure. In addition, ethanol synthesis only needs one native enzyme, and the hosts (e.g., *Saccharomyces cerevisiae*) are naturally tolerant to alcohols. The entire ethanol synthesis pathway is always inside of the cytosol, and thus they do not have mitochondrial transport limitations. These advantages explain why ethanol fermentation (over 100 g ethanol/L) is superior to any other biofuel processes.

Second, energy metabolism may become a critical issue for synthesizing fatty acid-based compounds, which are susceptible to changes in P/O ratio, ATP maintenance loss, and oxygen uptake fluxes. Comparing to alcohol production, fatty acid based fuels (such as biodiesel) requires longer biosynthetic pathways (more enzymes to overexpress) and considerable ATP usage for product synthesis (Steen *et al.* 2010). Besides, many enzymes in fatty acid pathway are tightly regulated during cell growth, leading to growth associated bio-production. The simultaneous biomass growth and fatty acid synthesis further exaggerates ATP shortage (He *et al.*
Therefore, aerobic fermentation has to be performed to enhance energy metabolism, which reduces product yield and increases the fermentation costs for aeration. Furthermore, the accumulation of fatty acid damages cell membrane and reduces oxidative phosphorylation efficiency. To demonstrate these synergistic effects on fatty acid yields, Figure 3.2c and d simulate E. coli fatty acid yields under different P/O ratios and metabolic burdens. As shown in Figure 3.2c, fatty acid production can achieve a similar yield as ethanol if the host’s biomass growth rate (as 0.05 h\(^{-1}\)) and energy maintenance is not high. In reality, fatty acid yield can drop to 50% or less of the theoretical maximum, which is in consistent with the model prediction if we considered a practical biomass growth, extra ATP maintenance, and low P/O ratio (< 1.5) in FBA (blue star in Figure 3.2d) (Orth and Palsson 2012). Figure 3.2d also indicates the high sensitivity of fatty acid yield in response to the P/O ratio (red star in Figure 3.2d). For instance, one unit change in P/O ratio leads to an abrupt drop in fatty acid yield -- from a theoretical maximum to zero.

3.6. Yin-Yang theory in metabolic engineering

To better understand the limitations of microbial cell factories, we refer to an ancient Chinese wisdom: Yin-Yang. Yin-Yang describes both the bright side and dark side of an object in the world. Yin and Yang oppose each other but are also interdependent. In the case of metabolic engineering, the microbial metabolism is operated by thousands of enzymatic reactions and mass transport processes that involve both carbon (Yang) and energy (Yin) transformations (Figure 3.1). Through billions of years of evolution and environmental adaptations, biological systems have evolved closely interdependent carbon fluxes for biomass growth and energy fitness, which are similar to the intertwined Yin-Yang forces. Although it is easy to engineer microbial hosts to produce small amounts of diverse products, manufacturing a particular compound with titers and
rates beyond the economic break-even point is difficult. In microbial conversions of a substrate to a product, metabolic entropy always increases if more carbon flux is redirected to the final products (Figure 3.3a & b). Figure 3.3c calculates the energy loss during conversion of glucose to biofuels at their theoretical yields as well as the practical yields. Based on the stoichiometry, theoretical energy losses during the conversions of glucose to biofuel molecules (alcohols and fatty acids) are small. However, much bigger losses are observed in real cases because of the suboptimal energy metabolism in biological systems.

To leverage the “Yin-Yang” balance, early metabolic engineers tried a few effective approaches to promote energy metabolism and boost productivity. For instance, Vitreoscilla hemoglobin (VHb), a soluble bacterial protein, has been used to enhance energy metabolism by promoting oxygen delivery, which can significantly improve cell growth and enhance chemical production under oxygen-limited conditions (Wei and Chen 2008). Furthermore, an energy-conserving pathway in E. coli was developed through metabolic evolution for high production of succinate from glucose fermentation (Zhang et al. 2009). This study indicates that the overexpression of a phosphoenolpyruvate carboxykinase increases the net production of ATP, compared to the primary mixed acid fermentation pathway via PEP carboxylase. The extra energy supply allows E. coli to produce succinate close to the theoretical maximum. In another case, an ATP-consuming reaction was introduced into S. elongatus PCC 7942 to drive carbon flux from acetyl-CoA to 1-butanol (Lan and Liao 2012). This study of 1-butanol production further validates that the ATP coupling reaction can make engineered pathways thermodynamically more favorable. To this end, we summarize the following suggestions to overcome the energy roadblocks.
First, a clear understanding of the entire carbon and energy metabolisms in microbial species would help us to conquer the energy limitations. Using \textit{E. coli} as an example, ATP significantly impacts the product distributions at the pyruvate node (Wang \textit{et al.} 2010a). Understanding ATP fluxes can offer rational design of \textit{E. coli} strains for improving product biosynthesis (Zhang \textit{et al.} 2009; Causey \textit{et al.} 2003). Flux balance analysis (FBA) and $^{13}$C-metabolic flux analysis (MFA) are the only available tools that can quantify energy expenditures. FBA can characterize cell energy metabolism by dividing ATP cost into non-growth associated loss and growth-associated maintenance (Varma and Palsson 1994). Due to the metabolic nature of suboptimal carbon fluxes, FBA, relying on the objective functions, may overestimate the cell potential for biosynthesis capability. $^{13}$C-MFA uses tracer experiments to constrain the FBA model so that it can precisely measure enzyme reaction rates. $^{13}$C-MFA can profile carbon fluxes through all energy generation/consumption pathways and deduce energy flows in the cell metabolism (ATP and cofactor balancing) (He \textit{et al.} 2014). Flux analysis not only allows us to determine the hidden Yin-Yang balance and to design rational engineering strategies, but also to characterize metabolic entropy and identify a strain’s energy potential for further improvement. Although $^{13}$C-MFA has not been widely accepted as a routine laboratory measurement tool to assess the engineered microbial hosts, this technology has excellent potential to reveal pathway engineering burdens (i.e., predict “the last straw” in genetic modifications). This tool can informatively tell metabolic engineers and project sponsors what can be done and what cannot be done.

Second, metabolic engineers need to exploit native pathways and avoid extensive pathway reconstruction. In history, many industrial successful cases of improved strain tolerance or productivity just relied on random mutation or evolution, leveraging Natural Selection of mutants for the best balance of ‘Yin-Yang’. Additionally, efforts should aim product synthesis at
pathways that do not require significant ATP expenditures (such as ethanol or organic acids). For example, the acetate overproduction pathway in \textit{E. coli} generates abundant ATP, and the engineered strain performs very well even when its oxidative phosphorylation, TCA cycle and competing fermentation pathways are disrupted (Causey \textit{et al.} 2003). When microbial hosts have low-burden biosynthesis pathways, they show robustness in industrial processes. Moreover, artificial synthetic circuits, efflux pumps, or novel pathways should be carefully considered in terms of the energy penalty. By revealing the tradeoffs behind synthetic biology parts via flux analysis approach, engineering strategies can be rationally designed.

Thirdly, although it is difficult to break the Yin-Yang balance in a natural microorganism, synthetic biologists may re-program the carbon metabolism and energy “fitness” by engineering novel microbial systems. Metabolic engineers often apply gene deletions, evolutionary engineering, or pathway overexpression to improve the strain productivity. These practices typically encounter adverse metabolic shifts due to energy imbalances. However, the creation of a “minimal or smart” cell can remove unnecessary genes in microbial hosts in effort to reduce cell burden and unlock the biosynthesis regulations (Forster and Church 2006; Trinh \textit{et al.} 2008). Additionally, synthetic biologists try to design and assemble minimal cells using synthetic chromosomes (Gibson \textit{et al.} 2010). These artificial biological systems do not necessarily follow the natural Yin-Yang balance evolved over billions of years, so they may have an unusually efficient energy metabolism, and thus achieve product yields close to the theoretical maximum.

Fourth, biological conversion can be integrated with non-living processes to reduce the biosynthesis burden. We can use robust microbial hosts to make simple molecules with high yields and titers, and then convert these molecules into a desired product with a complicated structure via biological and chemical processes. For example, the Keasling Lab achieved the
total synthesis of artemisinin with a two-stage semi-synthetic approach. They used the mevalonate pathway in *Saccharomyces cerevisiae* to synthesize artemisinic acid, followed by a four-step chemical conversion of artemisinic acid to artemisinin (Paddon and Keasling 2014). The Zhang Lab has made biopolymers by using engineered *E. coli* as a first step, to produce a simple molecule mevalonic acid, and then chemically converting it into biopolymers (Xiong *et al.* 2014). A significant advantage of these integrated processes is an extremely efficient bioconversion first step using a short pathway (Colletti *et al.* 2011). For instance, the titer of the semi-product mevalonic acid can reach as high as 88 g/L because its synthesis only requires three steps from the central metabolic node (acetyl-CoA) (Xiong *et al.* 2014). In another and more radical approach, an artificial cell-free system containing enzyme cocktails can mimic one or many functions of a biological system. Such systems can be used to synthesize products with near maximum theoretical yields (Hodgman and Jewett 2012; Ye *et al.* 2009). Cell-free systems can be designed to achieve optimal biosynthesis without cell maintenance cost.

Lastly, development of non-model microbial workhorses with desired traits in energy metabolism may achieve higher biosynthesis potentials, enabling the design of industrial biorefineries for the production of a broad range of products. In fact, even in the modern era of genomics, it is estimated that > 99% of all bacterial species remain unknown (Lasken and McLean 2014). Some non-model species might have a unique energetics that can facilitate product synthesis. For example, Algenol develops the engineered cyanobacteria for phototrophic ethanol production from CO$_2$ (http://www.algenol.com/). Moreover, cyanobacterial species have shown faster growth and higher production rate/titer by co-utilization of organic substrates (You *et al.* 2014; Atsumi *et al.* 2009). Cyanobacterial photo-fermentations, may facilitate cost-effective and large-scale biorefineries by using cheap feedstocks, CO$_2$, and light energy. In fact,
Nature is the best synthetic biologist and may have already prepared us an excellent chassis that we have not discovered yet. When we try to out-do Nature’s performance, we must first assimilate her lessons of ‘Yin-Yang’.

3.7. Conclusions

We have discussed the Yin-Yang concept as the underlying regulatory mechanism in cell metabolism. Biosynthesis of diverse useful products requires sophisticated genetic pathway engineering to steer a high flux to the final product while energy fitness requires the cell metabolism to be minimally changed. Since the powerhouse in microbial cell factory is not limitless, energy shortage eventually leads to metabolic shifts and reduced cell productivity in engineered microbes. The Yin-Yang balance may caution against the assumption that the host metabolism can be modified extensively to produce any desired products. By using fluxomics, we can formulate guidelines to avoid many false starts and dead ends during metabolic engineering. In addition, industrial bioprocess always faces numerous constraints and trade-offs (mass transfer limitations in fermentation, sterilization, strain stability, contaminations, and aeration costs). Feedstock selections, downstream product separation, and waste treatment are critical issues that impact product profitability. Thus, the design-build-test-learn cycle should cover both strain development and economic analysis. Nevertheless, the Yin-Yang philosophy provides general insights into biotechnology tradeoffs.

3.8. References


50. Lasken RS, McLean JS: Recent advances in genomic DNA sequencing of microbial species from single cells. *Nat Rev Genet* 2014, 15:577-584.


Figure 3.1. Cell carbon and energy metabolism illustrated by Yin-Yang Theory (note: engineered parts include plasmids, over-expressed enzymes, synthetic circuits, etc.)
Figure 3.2a *E. coli* strains producing ethanol (growth rate = 0.05 h\(^{-1}\))

Figure 3.2b *E. coli* strains producing isobutanol (growth rate = 0.05 h\(^{-1}\))
Figure 3.2c *E. coli* strains producing fatty acid (growth rate = 0.05 h⁻¹)

Figure 3.2d *E. coli* strains producing fatty acid (growth rate = 0.20 h⁻¹)
Figure 3.2a-d. Genome-scale FBA models for microbial biofuel mole-carbon yields from glucose. We use an *E. coli* FBA model (iJO1366) to predict production of different biofuels from glucose. Alcohol production is simulated under the microaerobic condition ($O_2$ influx $\leq 1.85$ mmol/(gDW·h)), while fatty acid is under aerobic condition ($O_2$ influx $\leq 12$ mmol/(gDW·h)). The medium conditions and glucose uptake rate (8 mmol/(gDW·h)) are same for all FBAs. Extra metabolic burden includes both protein overexpression and maintenance energy increase. Here, 10% extra metabolic burden is equivalent to 10% overexpression of biomass protein plus a proportional increase of non-growth associated ATP loss by 10 mmol ATP/(gDW·h). For each case, the objective function is set as to maximize the biofuel production. Abbreviations: DW (Dry Weight); FA (Fatty acid); Glc (Glucose); IB (Isobutanol).
Figure 3.3 Energy fitness and productivities in microbial cell factories.

Figure 3.3a: The trend of metabolic entropy changes (unit: ATP generation per glucose). In optimal metabolism, one mole of glucose generates 38 ATP for biosynthesis. Under constraints of P/O ratios and maintenance loss, less ATP can be generated (i.e., increase of metabolic entropy).

Figure 3.3b: The transition from carbon limitation to energy limitation with the increase of product yield. In many cases, the energy limitation prevents strains from achieving the yield and titer at break-even point.
Figure 3.3c: Cascade of energy changes (Heat of combustion) during biofuel synthesis from glucose. Reported yields: ethanol -- 96% of theoretical yield (Alper et al. 2006), isobutanol -- 85% of theoretical yield (Atsumi et al. 2008a), fatty acid --50% of theoretical yield (He et al. 2014), and H₂ (dark fermentation) -- 50% of theoretical yield (Rachman et al. 1997).
CHAPTER FOUR

ENHANCE ENERGY STATE OF FATTY ACID PRODUCING STRAINS

WITH VITREOSCILLA HEMOGLOBIN

4.1. Abstract
Engineered microbial species provide a sustainable platform to produce a wide range of chemicals from renewable resources. However, production of those compounds imposes significant metabolic burdens on host cells, leading to shifted metabolism, disrupted membrane, and unstable phenotype; those effects become even significant when oxygen becomes limited in the cell culture. Heterogeneous expression of Vitreoscilla hemoglobin (VHb) is known to enhance growth and energy efficiency of various hosts under microaerobic condition. In this study, we engineered fatty acids producing E. coli strain by introducing VHb and its mutant (VHb50), to solve the intracellular energy crisis. Growth and fatty production experiments indicated that the strain with VHb50 (strain GW50) achieved higher cell density and increased titer of fatty acids (50% improvement). In contrast, the benefit from wild-type VHb expression (GW1) counteracted its metabolic burden, and there is no significant difference in biomass and fatty acid titer. Further, expression of VHb50 significantly increased the ratio of unsaturated fatty acid (C16:1 and C18:1), especially oleic acid (C18:1), compared with the control strain without VHb. Lastly, we integrated the effect of VHb into flux models to simulate the responses of different host strains. The results demonstrated a different level of trade-off between the burden and the benefit from introduced genetic components, indicating the importance of specific properties of each genetic part.

4.2. Introduction
Metabolic engineering aims to obtain desired products through genetic modifications (Bailey 1991). The common approaches of dragging carbon fluxes to target molecules include gene knockout and heterogeneous enzyme expression. Even before the birth of ‘Metabolic engineering’, the fact that metabolic burden contributed from heterogeneous protein expression and plasmid maintenance led to decreased growth and shifted metabolism have been realized by researchers (Schaff et al. 1989; Birnbaum and Bailey 1991). With the advancement of DNA technologies (e.g., PCR (Saiki et al. 1985), compatible multiple plasmids system (Lutz and Bujard 1997), convenient genome DNA knockout (Datsenko and Wanner 2000), and advanced sequencing) as well as the increased demand of bulk chemicals production from green approaches, metabolic engineering have greatly extended its range of product to amino acids, drugs, polymers, and most recently biofuels(Stephanopoulos et al. 1998). After more than twenty years of development, the competition between introduced genetic parts and desired products for carbon and energy source became even intense, leading to unstable genotypes and phenotypes (production performance), and this situation became especially severe in engineered E. coli strains producing biofuels (Hollinshead et al. 2014; Wu et al. 2015; He et al. 2014; Lennen et al. 2011).

Fatty acids are important precursors for productions of biodiesel, surfactants, and lubricants in the industry. Biosynthesis of fatty acids or related compounds through metabolic engineering or synthetic biology has been a hot field during recent years (Jones et al. 2015). A series of approaches have been employed to improve fatty acid or biodiesel production including introduction of heterogeneous enzymes (Lu et al. 2008), knockout degradation pathway (Lu et al. 2008; Steen et al. 2010), reversal of degradation pathway (Dellomonaco et al. 2011), boosting regulatory factors that activate the fatty acid pathway (Zhang et al. 2012a), and employment of
dynamic sensor control system to relieve toxicity from intermediates (Zhang et al. 2012b; Xu et al. 2014). However, those high-yield fatty acid producing strains have been found to be unstable even at laboratory conditions (He et al. 2014). This situation became worse during process scale-up when local culture conditions (e.g., pH, oxygen, toxic compounds) are unfavorable. $^{13}\text{C}$-MFA has been applied to study the central metabolism of fatty acid producing strains, and the results indicated that remarkably high cellular maintenance energy was required for engineered strains producing fatty acid (He et al. 2014). Fatty acid production flux was found to be sensitive to P/O ratio as well as oxygen flux, based on FBA simulation (Wu et al. 2015). *Vitreoscilla* hemoglobin (VHb) has been well known to promote oxygen uptake and ATP production under oxygen limited condition in many hosts including *E. coli* (Khosla and Bailey 1988a). The introduction of VHb into fatty acid producing *E. coli* strains may potentially solve the problem of intracellular oxygen and energy limitation.

*Vitreoscilla* hemoglobin was first discovered by Webster and Hackett as early as 1966 (Webster and Hackett 1966). It was not realized as the first bacteria hemoglobin until its amino acid sequence was determined and showed high homology with eukaryotic hemoglobin (Wakabayashi et al. 1986). Heterogeneous expression of active VHb in *E. coli* was achieved two years later by two groups (Khosla and Bailey 1988a; Khosla and Bailey 1988b; Dikshit and Webster 1988). For the first time, Khosla and Bailey demonstrated that expression of VHb promoted oxygen uptake and improved the growth of host under microaerobic conditions. Later work by the same group further realized that VHb expression was able to enhance protein expression in *E. coli* under oxygen limited condition (Khosla et al. 1990). Considering of limited oxygen availability within cell culture at late exponential phase, the capability of VHb in enhancing growth and metabolites production have been widely applied to enhance production of
a wide range of compounds/proteins in various species (Zhang et al. 2007; Stark et al. 2015; Wei and Chen 2008). Those successful cases include: improved yields of cephalosporin C in *Acremonium chrysogenum* (DeModena et al. 1993), production improvement of human tissue plasminogen activator (tPA) in recombinant Chinese hamster ovary (CHO) cells (Pendse and Bailey 1994), promoted secretion/production of α-amylase and neutral protease in *B. subtilis* (Kallio and Bailey 1996), increased biomass weight, chlorophyll, nicotine, and reduced germination time (half time) in transgenic *Nicotiana tabaccum* (tobacco) (Holmberg et al. 1997), enhanced production rate and titer of erythromycin in engineered *Saccharopolyspora erythraea* (Minas et al. 1998). Several review papers have been published to summarize those successful applications of VHb in protein production and metabolic engineering (Zhang et al. 2007; Stark et al. 2015; Bülow et al. 1999; Frey and Kallio 2003; Stark et al. 2011).

Considering its competency and numerous successful applications of VHb, we proposed the hypothesis that VHb may also relieve the metabolic burden and intracellular stress caused by fatty acid production. To verify our hypothesis, we inserted VHb and its mutant into engineered *E. coli* strains that produced fatty acid. We also employed flux balance model to simulate the effects of VHb on fatty acid production.

4.3. Experimental and Methods

4.3.1. Chemicals and Strains

All chemicals were reagent grade and purchased from either Sigma-Aldrich (St. Louis, MO, USA) or Fisher Scientific (Pittsburgh, PA, USA) unless otherwise noted. Restriction enzymes, Phusion DNA polymerase, and T4 ligase were from New England Biolabs (Ipswich, MA, USA).
The DNA clean and concentrating kit, Gel Recovery kit, and Miniprep kit were from Promega (Madison, WI, USA).

DNA sequence of \(vhb\) and its mutant \(vhb50\) was based on previous report (Andersson et al. 2000). Both genes were synthesized by GenScript Inc. (NJ, USA) and cloned into pUC57 vector. \(E. coli\) DH10B was used for plasmid manipulation. The fadE knockout \(E. coli\) DH1 strain (endA1 recA1 gyrA96 thi-1 glnV44 relA1 hsdR17 (rK\(^-\) mK\(^+\) \(\lambda\)) and the plasmid pA58c-TR were from Dr. Fuzhong Zhang’s lab (He et al. 2014).

### 4.3.2. Plasmid construction

Primers used in this study were synthesized from Integrated DNA Technologies (Coralville, IA) and the detailed sequences are listed in Table 4.1. Plasmids GW1 and GW50 were constructed based on the plasmid pA58c-TR, to insert \(vhb\) or \(vhb50\) gene to the downstream of \(tesA\) gene. To ensure the expression of \(vhb/vhb50\) gene under the control of pLacUV5 promoter, an RBS sequence was added upstream of \(vhb/vhb50\) gene. (Figure 4.1) The \(vhb/vhb50\) genes were amplified using the primers \(Vh\_f\) and \(Vh\_f\), while the vector pA58c-TR was amplified by the primers of pA58C\_f and pA58C\_r (as shown in Figure 4.2). The PCR products were cleaned by DNA clean and concentrating kit (Promega) and then digested by XhoI and HindIII at 37 °C for two hours. The digested DNA fragments were purified through gel purification and ligated though quick ligation kit at room temperature for 10 min. Ligation products were subsequently transformed into DH10B chemical competent cells. After incubation overnight at 37 °C, colony PCR was employed to identify positive clones. A few positive clones were incubated in Luria-Bertani (LB) medium (37 °C 220 rpm) supplied with appropriate antibiotics (30 μg/mL chloramphenicol) overnight for Miniprep. DNA sequences of both plasmids were validated by sequencing services in Genome center at Washington University School of Medicine.
4.3.3 Medium and culture conditions

A M9 MOPS minimal medium was employed in this study for fatty acid production experiments (Neidhardt et al. 1974). The detailed composition of this medium is as following (per liter): 20 g of glucose, 6.8 g of Na₂HPO₄, 3.0 g of KH₂PO₄, 3.96 g of (NH₄)₂SO₄, 0.58 g of NaCl, supplemented with 50 ml of 1.5 M MOPS with pH adjusted to 7.4 with KOH, 1 ml of 1 M MgSO₄, 0.1 ml of 10 mg/ml vitamin B1, 1 ml of 0.1 M CaCl₂, 1 ml of 1000X micronutrients solution including 1.6 g of MnCl₂·4H₂O, 2.9 g of ZnSO₄·7H₂O, 2.5 g of H₃BO₃, 2.8 g of FeSO₄·7H₂O, 0.71 g of CoCl₂·6H₂O, 0.48 g of CuSO₄·5H₂O, and 0.37 g of (NH₄)₆Mo₇O₂₄·4H₂O.

For fatty acid production in minimal medium, a single colony of cells from a fresh plate was used to inoculate a 5 ml of LB media for the pre-culture. The pre-culture was grown overnight at 37°C on a rotary shaker at 225 rpm. The pre-culture was used to inoculate (2%, v/v) a 5 ml of minimal medium for overnight growth. After reaching stationary phase, the first minimal culture was subsequently inoculated into the second minimal medium (0.5%, v/v) grown on a rotary shaker in 25 ml tubes (25 x 150 mm) at 37°C and 225 rpm. IPTG was added at appropriate concentrations when cell growth reaches an early exponential phase (OD₆₀₀ ~ 0.8), to induce gene expression under the control of P_LacUV5 promoter. The liquid cultures from each tube were centrifuged and the supernatant was separated from the biomass. Both the biomass and the supernatant samples were stored in -20 °C prior to analysis.

4.3.4 Fatty acid measurement

200 µl of cell culture was mixed with 300 µl of dH₂O, and get acidified using 50 µl of concentrated HCl. 500 µl of EtAc spiked with C19:0 ME as internal standard was added to extract fatty acid from water phase. 400 µl of organic phase containing fatty acids separated by
centrifugation was transferred to a new tube and another 500 µl of EtAc was added again to repeat the extraction. 100 µl of MeOH:HCl (9:1) was added to the EtAc extract and mixed well. 100 µl of TMS-diazomethane (2 M in hexanes) was also added under hood and the reaction was kept in the hood for 10 ~ 15 min at room temperature. The methyl esters of fatty acids were analyzed using a gas chromatograph mass spectrometer (GC-MS) (Hewlett Packard 7890A and 5975C, Agilent Technologies) equipped with a DB5-MS column (J&W Scientific). The GC-MS program was as follows: the column temperature initially held at 80 º C for 1 min, raised to 280 º C at 30 º C·min⁻¹ and held at 280 º C for 3 min. Helium was used as the carrier gas. The mass spectra were analyzed using the Enhanced Data Analysis software (Agilent Technologies). The fatty acids were quantified based on the standard curve of standard mixtures of methyl esters of fatty acid.

4.3.5. Simulate cellular physiologies with flux balance model

Genome scale model iEcDH1_1363 (includes 1363 genes, 2752 reactions, and 1950 metabolites) was employed to simulate fatty acids production in DH1 strain (Monk et al. 2013). A simplified flux of fatty acid (C16:0) was added as representative of fatty acid production, and the objective function was set to maximize this flux (Wu et al. 2015). The growth rate of engineered strain was set as 0.2 h⁻¹, considering that fatty acid production is growth associated (He et al. 2014; Lu et al. 2008). Default values were employed for the boundary of all fluxes except the followings: The upper and lower boundary of flux 1032 was set to be zero for ΔfadE; the sensitivity of maintenance energy was tested through FBA; the lower boundary of glucose uptake flux was set as experimental value. We assumed VHb improved the affinity of terminal oxidase (i.e.,
cytochrome $d$ or cytochrome $o$), leading to increase uptake flux of glucose. We also assumed that VHb had no influence on glucose uptake based on previous reports (Tsai et al. 1996a).

The COBRA toolbox and LibSBML library were employed for genome scale model manipulation (Schellenberger et al. 2011; Bornstein et al. 2008); while Gurobi 5.5 linear solver (Gurobi Optimization Inc.) was utilized for FBA calculation on MATLAB 2012b.

4.4. Results and discussion
4.4.1 Growth kinetics and fatty acid production
Growth kinetics for all strains incubated in M9 minimal medium was described in Figure 4.6. Shaking tubes were employed here for obtaining microaerobic conditions. There is an obvious exponential phase rate for each strain after IPTG induction. The growth rate of exponential phase was very close to the previous report: pA58c 0.29 h$^{-1}$, GW1 0.19 h$^{-1}$, GW50 0.29 h$^{-1}$ (He et al. 2014). Introduction of VHb only showed apparent improvement during late exponential phase of GW50.

The trend of fatty acid production is quite similar to the biomass growth (shown in Figure 4.6): after 24 hr of induction, GW50 produced most fatty acid, while GW1 produced least fatty acid. Fatty acid titer of the control strain pA58c in this study is much lower than previous report by our lab (He et al. 2014), mostly due to poor oxygen supply in shaking tubes compared with baffled shaking flasks. The importance of oxygen on cell growth and free fatty acid production can also be revealed from this difference.

To interpret the profiles in biomass growth and fatty acid production, two factors may contribute most: First, wild-type VHb did improve oxygen uptake and energy metabolism in GW1 strain; however, this benefit did not counteract the metabolic burden caused by VHb. This
observation reflected the importance of specific properties for each individual cellular component, where protein engineering (e.g., direct evolution or rational design) is able to improve/change catalytic kinetics of single enzyme, leading improved yield of desired products (Leonard et al. 2010; Bommareddy et al. 2014). On the other side, the impact of VHb50 was only significant when cell density was relatively high and oxygen condition was poor, agreeing well with all previous reports regarding VHb (Zhang et al. 2007; Tsai et al. 1996b). This nature actually limits further application of VHb is, because industrial production always requires high production rate where fully aerobic condition is employed. To resolve this bottleneck, a deep understanding of VHb mechanism is necessary.

The role of VHb has been considered to be improving oxygen transfer, boosting ATP generation, and promoting intracellular energy state when extracellular oxygen concentration is low (Tsai et al. 1996a; Tsai et al. 1996b; Kallio et al. 1994). Compared with other hemoglobin, purified VHb has a medium affinity for oxygen binding as well as relatively slow rate for oxygen release in vitro (Giangiacomo et al. 2001), implying its possible function as assisting oxygen transport rather than transporter (structure of active VHb shown in Figure 4.2). However, later work proved that lipid-bound VHb has a significantly higher affinity with oxygen (20 folder enhancement), suggesting its potential role in oxygen transport as a membrane protein (Rinaldi et al. 2006). Other functions of VHb have also been reported as relieving oxidative stress, detoxifying nitric oxide (NO) in vivo (Frey et al. 2002), and enhancing intracellular level of tRNA and ribosome (Roos et al. 2002). Further work will provide more insights over detailed mechanism of VHb.

4.4.2 Expression of VHb affects the degree of unsaturation of free fatty acid
The profile of fatty acid also experienced a significant shift after the introduction of VHb (shown in Figure 4.6). At both early exponential (5 hr after IPTG induction) and late exponential phase (24 hr after IPTG induction), the percentage of unsaturated fatty acid (C16:1 and C18:1) increased significantly (from 0.37 to 0.46), especially for the oleic acid (C18:1). Cao et al. has reported that simultaneous overexpression of fabA and fabB will increase the ratio of unsaturated fatty acid (Cao et al. 2010). In another study, expression of FadR was also found to enhance the percentage of unsaturated fatty acid by activating expressions of fabA and fabB (Zhang et al. 2012a). However, heterogeneous expression of VHb in Aurantiochytrium sp. led to a decreased percentage of unsaturated fatty acids (Suen et al. 2014). Previous studies also proved that VHb was able to boost the intracellular expression of heterogeneous proteins in E. coli (Khosla et al. 1990). Based on those facts, we infer that VHb may promote the leaky expression of fadR in plasmid, leading to elevated level of fabA and fabB pathway. To further confirm this effect, evidence from transcriptional or proteomics level will be helpful.

Increased degree of unsaturation of fatty acid is desired for several reasons. One important fact is that biodiesel derived from high content of unsaturated fatty acids have lower melting temperature, which is suitable as fuels for cold environments (e.g. winter). Another feature is that uptake monounsaturated fatty acid may raise the level of high-density lipoprotein (HDL) cholesterol, which is considered as ‘good’ cholesterol. Introduction of VHb into algae may facilitate its production of omega-3 oil.

4.4.3. Effect of oxygen and maintenance energy on fatty acid production

To quantify the effect of oxygen and maintenance energy on fatty acid production, FBA simulation was employed as described in the section of methods. To simulate cellular
physiologies at different growth phases, we adopted two set of conditions describing early exponential phase ($v_{\text{glucose}} = 7.6 \text{ mmol/gCDW\cdot h}$, growth rate $\mu = 0.29 \text{ h}^{-1}$, default P/O ratio = 1.75) and late exponential phase ($v_{\text{glucose}} = 4 \text{ mmol/gCDW\cdot h}$, growth rate $\mu = 0.1 \text{ h}^{-1}$, deceased P/O ratio = 1). These parameters were either from previous report or experimental measurement in this study (He et al. 2014). Considering of membrane disruption and proton leakage caused by free fatty acid accumulation (Lennen et al. 2011), we assumed a decreased P/O ratio as 1 for the late exponential phase (Heyland et al. 2011).

The simulation results were shown in Figure 4.7a (for early exponential phase) and Figure 4.7b (for late exponential phase). At early exponential phase, theoretic maximal yield of fatty acid was $\sim 0.24 \text{ g FA/g glucose}$ (for growth rate $\mu = 0.29 \text{ h}^{-1}$, 0.256 g FA/g glucose for growth rate $\mu = 0.26 \text{ h}^{-1}$). In our previous report, the highest yield was 0.17 g FA/g glucose, which is already 66% of the theoretic value (He et al. 2014). Taken cellular maintenance energy of the fatty acid strain into consideration, 0.17 g FA/g glucose would be very close to the maximal yield at this condition. On the other side, a rough estimation of cellular maintenance energy can be made based on FBA prediction: The maximal maintenance energy for control strain pA58c was 30 mmol/gCDW\cdot h in the minimal medium when oxygen supply is sufficient. Compared with other reports on maintenance energy for engineered strains (Heyland et al. 2011), that value is quite reasonable. Another obvious trend was that with the decrease of oxygen flux and the increase of maintenance energy to certain level, the yield of fatty acid would meet a sudden drop – ‘energy cliff’ as we discussed in our previous report (Wu et al. 2015). At this period, a slight improvement in oxygen flux (e.g., from 14 to 16 mmol/gCDW\cdot h, assumed enhancement by VHb) would not make any significant difference in fatty acid yield. In late exponential phase, when cell culture reaches a relatively high density, the oxygen condition in medium becomes
microaerobic (as shown in Figure 4.7b). A minor increase in oxygen flux (e.g., from 4 to 6 mmol/gCDW·h, assumed enhancement by VHb) will harvest a significant improvement in fatty acid yield. That also explained why the effect of VHb was significant at microaerobic conditions. Notably, the expression of VHb from a medium copy of plasmid would bring a certain amount of metabolic burden. This metabolic burden is not too much, however, for fatty acid producing strain with high cellular maintenance energy, this additional burden may become ‘the straw that broke the camel’s back’ if oxygen is also limited. There is no report regarding how the structure difference (i.e., amino acid mutations) leads to the function improvement in VHb50 (Andersson et al. 2000), compared with WT VHb. From FBA simulation, we infer the expression of VHb50 may make a larger improvement in oxygen uptake flux, compared with WT VHb.

4.5. Conclusion

In this study, our hypothesis was confirmed that the introduction of VHb was able to relieve the metabolic stress of fatty acid producing strain and improve the final titer of biomass and fatty acid. We further revealed that VHb expression would improve the ratio of unsaturated fatty acid, which may shed light to further application of VHb to produce fuel with lower melting point(Zhang et al. 2012a) or other valuable products with high degree of unsaturation. Lastly, we demonstrated the importance of individual VHb properties on the performance of engineered strains. To sum up, engineered components (e.g., enzyme, transporter, or circuit) would only bring expected enhancement when its benefit beats it burden.

4.6. Reference


<table>
<thead>
<tr>
<th>Name of primer</th>
<th>Sequence (5’—3’)</th>
</tr>
</thead>
<tbody>
<tr>
<td>pA58C_f</td>
<td>GCAC <strong>AAGCTT</strong> CCAGGCATCAATAAATAAAGCA A</td>
</tr>
<tr>
<td>pA58C_r</td>
<td>CCTTA <strong>CTCGAG</strong> TTATGAGTCATGATT TTACT</td>
</tr>
<tr>
<td>Vh_f</td>
<td>CG CAT <strong>CTCGAG</strong> TTTAAGAAGGAGATATACAT ATGTTAGACCAGCAAACCATA</td>
</tr>
<tr>
<td>Vh_r</td>
<td>GCAC <strong>AAGCTT</strong> TTATACCACCGCTTGAGCGTA</td>
</tr>
</tbody>
</table>

Table 4.1 DNA sequence of all primers used in this work

‘__’ indicates the restriction cutting sites.
Figure 4.1 Structure of *Vitreoscilla* hemoglobin in active dimer form, simulated by VMD
VHb gene and mutants (VHb50)*

Figure 4.2 Genetic manipulations to insert VHb into pA58c-TR
Figure 4.3 Optimization of IPTG concentrations for VHb50 expression

Figure 4.4 Growth curve for three fatty acid producing strains
Figure 4.5 Fatty acid productions after 24 hr of IPTG Induction
Figure 4.6a Free fatty acid production profile for control strain and VHb strain after 8 hr IPTG induction

Figure 4.6b Free fatty acid production profile for control strain and VHb strain after 24 hr IPTG induction
Figure 4.7a Effect of oxygen flux and maintenance energy on fatty acid yield at exponential phase

Figure 4.7b Effect of oxygen flux and maintenance energy on fatty acid yield at late exponential phase
CHAPTER FIVE

BUILD WEB-BASED PLATFORM FOR FLUXOMICS STUDIES:
MICROBESFLUX REBUILD AND WEBSITE DEVELOPMENT

5.1. Abstract

Metabolic flux analyses offer direct insights into cell metabolism. Metabolic flux analyses include genome-scale Flux Balance Analysis (FBA) and $^{13}$C-Metabolic Flux Analysis ($^{13}$C-MFA). To speed up fluxomics studies, we need a user-friendly platform to construct metabolic networks from genome information and perform flux calculations using $^{13}$C data. Four years ago (2011), Tang lab developed a web-based platform (MicrobesFlux) for reconstructing metabolic models from the KEGG database (Kyoto Encyclopedia of Genes and Genomes). The platform ran on a shared server system at Washington University in St. Louis. Unfortunately, this server was unstable and suffered from downtime occasionally. Hence, we set out to rebuild MicrobesFlux on a commercial server to make the systems more usable. The enhanced MicrobesFlux updates metabolic network information with the latest version from the KEGG database. In addition, we added MATLAB programs into the platform so that it can also provide $^{13}$C-MFA. The new program (called WUFlux, WU: Washington University) can carry out $^{13}$C-MFA of different metabolic types for prokaryote species. Furthermore, WUFlux also contains a carbon fate map of central pathways, labeling correction programs, and user manuals. The systems we developed are open-source and free to use. The new platform for fluxomics study is now available at http://www.fluxomics.net, and we will continue to improve the functionalities of our software for both FBA and $^{13}$C-MFA.
5.2. Introduction

Metabolic flux analysis is widely used to predict or measure *in vivo* enzyme reaction rates in microbes. FBA studies microbial metabolism based on the stoichiometry of the metabolic reactions as well as the measurement of inflow (substrate uptake) and outflow fluxes (biomass and product synthesis). FBA requires an objective function (e.g., optimization of biomass yield) to estimate the flux values. A more rigorous flux analysis combines the flux stoichiometry with $^{13}$C isotopic tracing, i.e., $^{13}$C-MFA. $^{13}$C-experiments consist of feeding the cell culture with a defined $^{13}$C-substrate to fingerprint downstream metabolites with the labeled carbon ($^{13}$C). The patterns of isotopic enrichment in metabolites, once $^{13}$C has reached a steady state distribution throughout the metabolic network, can be used to decipher flux distributions in the cell metabolism. The isotopomer information can discover novel pathways, resolve reversible and branched fluxes, and quantify circular metabolic routes (e.g., the TCA cycle). On the other hand, $^{13}$C-MFA, requiring both experimental and modeling efforts, is time-consuming and costly (Figure 5.1). In general, one $^{13}$C-MFA project can take several experienced researchers more than one year to accomplish (based on personal communications to $^{13}$C-MFA groups). Although $^{13}$C-MFA began as early as the 1980s, it has not been as widely used as other analytical/systems biology tools (Crown and Antoniewicz 2013). To date, published $^{13}$C-MFA papers are fewer than 1000 (Crown and Antoniewicz 2013). Based on Pubmed database, in 2012 and 2013, only 41 papers (related to $^{13}$C metabolic flux analysis) have been published. Among them, 18 were review or method papers, while 10 were research papers on bacteria and the remaining 13 papers were research on yeast or mammalian cells. The progress of flux analysis has been slow because the construction of a metabolic network based on an annotated genome, $^{13}$C-labeling tracing, and flux calculation can all be very time-consuming. On the other hand, the world’s bacterial species
number between $10^7$ to $10^9$, and ~$10^5$ species have been sequenced for their 16S rRNA genes (Schloss and Handelsman 2004). There are fewer than one hundred $^{13}$C-MFA studies for nonmodel species. The gap between $^{13}$C-MFA studies and sequenced bacterial species calls for broad-scope fluxomics tools for characterization of a large amount of unknown microbial species. To reduce modeling challenges, FBA platforms have been developed to facilitate reconstructing genome-scale metabolic networks. These platforms include SuBliMinal (Swainston et al. 2011), SEED (Henry et al. 2010), RAVEN (Agren et al. 2013), Pathway Tools (Karp et al. 2002), COBRA (Becker et al. 2007), and FAME (Boele et al. 2012). These platforms have been discussed in a recent review paper (Hamilton and Reed 2014). $^{13}$C-MFA software platforms are also being developed, including FiatFlux (Zamboni et al. 2005b), iMS2Flux (Poskar et al. 2012), INCA (Young 2014), Metran (Yoo et al. 2008), OpenFLUX (Quek et al. 2009a), OpenMebius (Kajihata et al. 2014), 13CFLUX (Wiechert et al. 2001) and 13CFLUX2 (Weitzel et al. 2012). With rapid advances in genome sequencing, network reconstructions and $^{13}$C-based functional analysis are concurrently required for metabolic characterizations.

To augment these tools, we built an integrated platform at www.fluxomics.net, combining the functions of both FBA -- MicrobesFlux, at (www.microbesflux.org, as shown in Figure 5.2) and $^{13}$C-MFA -- WUFlux, at (13cmfa.org, as shown in Figure 5.3). MicrobesFlux is a web platform to draft and reconstruct metabolic models from KEGG (www.genome.jp/kegg/). However, the test version of MicrobesFlux ran into problems on its shared server, which suffers from instability and occasional downtimes. To resolve this issue, we ported the system to the Amazon server EC2 (as shown in Figure 5.4) and rebuilt the genome modeling system. Our tool is a template-based package for tracking carbon transition and performing isotopomer corrections and $^{13}$C-MFA. WUFlux consists of three parts: a) a carbon transition map (CTM),
which is the basis for building $^{13}$C-MFA models; b) an isotopomer analysis package for analyzing both amino acids and key free metabolites in the central metabolism (based on a MATLAB program by the Wiechert group (Wahl et al. 2004)); c) and open source MATLAB program files which calculates $^{13}$C-MFA from MID information. Our websites can be accessed via the Internet and all fluxomics tools are free for use/download and easy to expand (for WUFlux). In summary, the new version of this web-based platform offers a programming-free and user-friendly broad scope tool that supports flux analysis studies.

5.3. Implementation

5.3.1. MicrobesFlux update

MicrobesFlux is a free open-source platform. Our previous paper (Feng et al. 2012) has described the details of this platform and provided a user manual. In a nutshell, MicrobesFlux consists of four parts: a web front-end (the user interface), a backend, a task processing system, and an optimization server. In the past, MicrobesFlux had issues with the host machine on which the task processing system and the optimization server were running. When we first built MicrobesFlux, there was no specific funding for the project. Therefore, we had to rely on a server provided by Washington University, which is shared by other users and frequently restarted, resulting in an unstable task processing system that can delay user-submitted tasks. Besides, we did not have access to the most up-to-date KEGG models due to KEGG’s paid subscription model. In the summer of 2014, we received a grant to continue work on this topic, fixing problems and updating our database versions. During the process of reloading MicrobesFlux, we moved the backend to a stable, commercial server and completely rewrote the task processing system. We also added a monitoring function to the platform to better manage the task processing system.
5.3.2. New features of reloaded MicrobesFlux

a. We have updated our backend KEGG database. Users can run FBA on any KEGG organisms as of September 2014. The new database in MicrobesFlux includes 3192 species, compared to 1304 species in the test version.

b. We now support the SBML output format, which broke down earlier.

c. Users can now store an unlimited number of models. Previously, we periodically purged models due to storage constraints. Now the backend, running on Amazon EC2, supports an unlimited number of models for any user.

d. We now have a robust task management system. Users are guaranteed to get their results back within 24 hours of submitting the optimization job.

5.3.3. Development of websites for fluxomics studies

We have built a comprehensive web-based platform including various tools (most tools were developed by our lab) for fluxomics research (as shown in Figure 5.2). In our website (fluxomics.net, as shown in Figure 5.2), users can read the latest publications by clicking the button ‘Enter 13C Flux News’; they can also build up and calculate their genome-scale model by entering MicrobesFlux; they can further get $^{13}$C-MFA tools by visiting WUFlux. The use of MicrobesFlux is kept the same as it was released four years ago. For WUFlux, users can download $^{13}$C-MFA tools: the carbon transition map (CTM), the $^{13}$C-MFA software package, and the MS correction Tool (as shown in Figure 5.3) by simply clicking corresponding labels and saving packages into their PCs.

5.4. Results

A comprehensive platform incorporating both functions of FBA and $^{13}$C-MFA has been released. The new version of MicrobesFlux has been tested thoroughly. Additionally, the newly
integrated WUFlux software has a user-friendly interface; all functions are easy to operate, and calculation can be saved at any stage. Users can choose different templates for various labeled substrates and metabolic networks. By implementing WUFlux, researchers without professional knowledge of $^{13}$C-MFA can easily get flux data of high quality from raw MS data. Because the MATLAB codes of all program files in WUFlux are open to researchers, users can extend or enhance its capability by editing the MATLAB program. Finally, WUFlux includes a carbon fate map and a labeling correction tool for amino acids and free metabolite analysis, which can facilitate future application of $^{13}$C-MFA. The completely open-source platform makes good feasibility for further development. We will continue to collect users’ feedback and improve its performance in the future. We hope that our platform can not only provide broad-scope fluxomics functions for characterization of novel microbial species, but also facilitate rational metabolic engineering.

### 5.5. Availability and requirements

- **Project name:** Fluxomics
- **Project homepage:** [http://fluxomics.net](http://fluxomics.net) (for each individual project, MicrobesFlux is at [http://microbesflux.org](http://microbesflux.org) and WUFlux is at [http://13cmfa.org](http://13cmfa.org))
- **Operating systems:** Platform independent
- **Programming language:** Java, Python and MATLAB (for 2012b and later version)
- **License:** Both MicrobesFlux and WUFlux are freely available.
- **Any restrictions to use by non-academics:** none
5.6. References


Figure 5.1. $^{13}$C-MFA protocol and sources of flux analysis variance; in general, a $^{13}$C-MFA requires months of work to accomplish. MFA errors (in blue boxes) can come from both experimental measurements and modeling calculations.
Figure 5.2. The webpage of our platform for comprehensive fluxomics studies (http://fluxomics.net).
Figure 5.3. The webpage of WUFlux (http://13cmfa.org), which can be accessed and freely download
Figure 5.4. The webpage of Amazon server EC2, (a) all Amazon web services, (b) buckets of our websites
CHAPTER SIX

RAPID PREDICTION OF BACTERIAL FLUXOMICS USING MACHINE LEARNING AND CONSTRAINT PROGRAMMING

6.1 Abstract

Metabolic flux reflects a functional aspect of cell physiology. $^{13}$C-MFA ($^{13}$C metabolic flux analysis) has been widely used to measure in vivo enzyme reaction rates (i.e., metabolic flux) in microorganisms. Mining the relationship between environmental and genetic factors and metabolic fluxes hidden in existing fluxomic data will lead to predictive models that can significantly accelerate flux quantification. In this paper, we present a web-based platform (MFlux: http://130.101.92.205/influx/) that predicts the bacterial central metabolism via machine learning, constraint programming, and quadratic programming, leveraging data from over 100 $^{13}$C-MFA papers on heterotrophic microbial metabolisms. Three machine learning methods, namely Support Vector Machine (SVM), k-Nearest Neighbors, and Decision Tree, were employed to study the sophisticated relationship between environmental and genetic factors and metabolic fluxes. We performed a grid search of the best parameter set for each tested algorithm and verified their performance through 10-fold cross validation. SVM yielded the highest accuracy of all three algorithms, with average error rate under 5%. Further, we employed quadratic programming to adjust flux profiles to satisfy stoichiometric constraints. Experimental results showed that MFlux can reasonably predict fluxomes as a function of bacterial species, substrate types, genetic modifications, growth rates, oxygen conditions, and cultivation methods.
6.2. Authors’ Summary

Metabolic information is important for disease treatment, bioprocess optimization, environmental remediation, biogeochemical cycle regulation, and our understanding of life’s origin and evolution. Fluxomics can quantify microbial physiology at the level of metabolic reaction rates. To speed up $^{13}$C-MFA, we hypothesize that genetic and environmental factors generate specific fluxome patterns that can be recognized by machine learning. Aided by constraint programming and quadratic optimization, our machine learning platform can predict meaningful metabolic information about bacterial species in their environments. Further, it can offer constraints to improve the accuracy of flux balance analysis. This study infers that the bacterial metabolic network has a certain degree of rigidity in allocating carbon fluxes, and different microbial species may share common regulatory strategies for balancing carbon and energy metabolisms. As a proof-of-principle, the use of data driven models (e.g., artificial intelligence) may assist mechanistic based models to elucidate the topology of microbial fluxomes.

6.3. Introduction

With the advent of systems biology tools such as genomics, transcriptomics, proteomics, and metabolomics during the last decade, the understanding of intracellular metabolisms from genotype to phenotype has been dramatically boosted. Notably, $^{13}$C-MFA enables the quantification of metabolic reaction rates in vivo [1]. It determines carbon metabolic fluxes using the mass isotopomer distribution (MID) of proteinogenic amino acids or free metabolites from $^{13}$C labeling experiments. $^{13}$C-MFA is considered as a reliable measurement of central metabolic
reaction rates [2], which has demonstrated its power in discovering novel pathways [3,4], validating gene functions [3], verifying engineered strains [5,6], and revealing energy metabolisms of host strains [7]. In the past decade, advanced parallel bioreactor systems, mass spectrometry, and computational tools resolving metabolic fluxes have been developed [8,9,10,11], which improved the precision of flux profiles [12] and extended $^{13}$C-MFA’s application to the non-stationary metabolic phase [13,14]. On the other hand, broad applications of $^{13}$C-MFA are still hindered because $^{13}$C-experiments, biomass analysis, and flux calculations are expensive and time-consuming [15]. Moreover, some microbial systems may not be amenable to $^{13}$C-MFA if they require complex nutrients or their genome annotation is incomplete [16]. Before performing $^{13}$C-MFA on non-model species, laborious work is needed to examine extracellular metabolites, to characterize unknown pathways, and to analyze biomass compositions.

This study aimed to employ an artificial intelligence (AI) approach called machine learning (ML) to investigate bacterial fluxomics patterns. ML is a powerful tool in systems biology [17] and has demonstrated successes in omics studies [18,19]. For example, the precision of genome annotation on the model species C. elegans has been enhanced by employing a simplified SVM (support vector machine) method. Researchers have reached an accuracy of 75% on controversial genes [20]. At the transcriptomics level, ML approaches are used for disease identification. For instance, SVM has successfully recognized the gene expression patterns of hepatocellular carcinoma [21], diffuse large B-cell lymphoma [22], and ovarian cancer [23]. At the proteomics level, Supek et al., have employed a combined approach by integrating the Principal Component Analysis (PCA) method with SVM, to enhance analytic power in identifying ‘fingerprint’ proteins (i.e., unique proteins in each tissue) from different horseradish
tissues (leaf, teratoma, and tumor) grown \textit{in vitro} \cite{24}. In metabolomics, an SVM method can resolve the NMR data of metabolites in urine samples from different groups of people (healthy vs. pneumonia) \cite{25}. In metabolic modeling, Karp’s group adopted ML algorithms to predict the existence of various pathways for metabolic network reconstruction in different organisms \cite{19}.

The general idea of ML is to statistically build a predictive \textit{model} or an \textit{estimator} $\mathbb{R}^n \rightarrow \mathbb{R}$ that maps an n-dimensional real number vector called the \textit{feature vector} to a real number called the \textit{target} or \textit{label}. If the target takes discrete values, we call the ML model a \textit{classifier}; otherwise, a \textit{regressor}. A pair of a feature vector and a target forms a \textit{sample}. Given samples, a machine learning algorithm will find such a mapping, usually through solving a numerical optimization problem, to minimize the predictive error. Samples used to train a model form the \textit{training set} while those for testing the performance form the \textit{testing set}. The models learned through ML are usually not analytical models that can be represented using an equation. Rather, they are numerical operators. For example, an artificial neural network (ANN) model can be represented by many matrices, and when being used to predict, the input variables will be multiplied with those matrices sequentially. A bad model can only predict well on the training set as if it ‘remembers’ the training samples, while a good model can learn the patterns among data and still be accurate on samples it has never ‘seen’. Hence, researchers usually make the training and test sets mutually exclusive. A mechanism called \textit{cross validation} is used to ensure the mutual exclusiveness of training and test sets while make full use of all data.

A distinct advantage of ML applications is that they can reduce the need for costly experimental supplies and time-consuming bench work. Despite the progress in utilizing ML methods in systems biology, there is no similar application in the fluxomics field to predict the flux profile. Therefore, we conceived the idea of integrating ML strategies with fluxomics
research. To efficiently employ machine learning methods, a database with a sufficient number of samples is a prerequisite. Recently, a $^{13}$C-MFA database (‘CeCaFDB’) has been constructed, which includes over 100 papers (mostly on prokaryotic metabolisms) [26]. Based on this database, we initiated five categorical and sixteen continuous features to describe the environmental and genetic factors involved in $^{13}$C-MFA of bacterial species. Unlike most omics projects employing ML approaches, this work built regression models rather than classifiers: Twenty-nine lumped central metabolic fluxes were adopted as the outputs to describe bacterial carbon metabolisms. A 10-fold cross validation evaluated the performance of different algorithms. Furthermore, we included a knowledge-based system to check whether user inputs were biologically meaningful. Lastly, quadratic programming was employed to adjust the fluxes predicted by ML to satisfy the stoichiometric constraints. Our web-based platform (‘MFlux’) provides reasonable predictions for central metabolic flux profiles on 30 bacterial species, and it can be accessed online (http://130.101.92.205/influx/). Although our platform is still in the early phase, our attempt to employ an AI approach in fluxome studies will have broad impacts on both systems biology and metabolic engineering fields.

6.4. Methods

6.4.1. Data collection and preprocessing

All the training data for MFlux comes from the literature. The total uptake rate of carbon sources is defined as 100; all other fluxes are normalized to a scale of 100. We obtained $^{13}$C-MFA information for bacterial species from the CeCaFDB database and added a few recent papers (total ~120 papers, as of January 2015) [26]. $^{13}$C-MFA data related to photosynthetic bacteria was excluded in this study because of their unique fluxome topologies (such as the Calvin Cycle and the reversed TCA cycle) and insufficient sampling sizes for ML.
In heterotrophic microorganisms, interconversions between glycolysis metabolites (phosphoenolpyruvate and pyruvate) and TCA cycle metabolites (oxaloacetate and malate) involve a set of anaplerotic reactions (e.g., phosphoenolpyruvate carboxylase, phosphoenolpyruvate carboxykinase, pyruvate carboxylase, and malic enzyme), serving as a key switch points for carbon flux distribution in bacteria [27]. These reactions, balancing both carbon and cofactors, may be employed by different microbial species. For example, *E. coli* anaplerotic pathways involve phosphoenolpyruvate carboxylase and malic enzyme, while *Bacillus* species furnish pyruvate carboxylase (the pyruvate shunt). In the case of *Corynebacterium*, both phosphoenolpyruvate carboxylase and pyruvate carboxylase are functional [28,29]. These anaplerotic pathways can re-route fluxes when a central pathway such as pyruvate kinase is knocked out. To ease the machine learning efforts, the anaplerotic pathways are lumped into two routes that exchanges fluxes between the TCA cycle and the glycolysis nodes: MAL $\leftrightarrow$ PYR + CO$_2$ and OAA $\leftrightarrow$ PEP + CO$_2$. This simplification also considered the fact that $^{13}$C-MFA has poor resolution on anaplerotic fluxes because various combinations of these reactions can generate similar labeling patterns in amino acids [30].

### 6.4.2. Feature selection and scaling

ML can make predictions with iteratively-tuned parameters and well-trained models to account for influential factors in cell metabolism. Based on published $^{13}$C-MFA methodologies and microbial physiologies, we proposed five categorical features: species, nutrient types, oxygen conditions, genetic background, and cultivation methods. We had two considerations during feature selection: First, genetic modifications can significantly re-organize fluxomes. To improve the predictability on mutant strains, our platform allows “turn-off” or “turn-on” certain central pathways (by manually setting the flux boundaries) in engineered strains. Second, the
factor of cultivation method aims to reveal fluxome differences between shake flask cultures (a pseudo-steady state approach) and bioreactor cultures (a well-controlled fermentation or chemostat cultivation). Meanwhile, we have sixteen continuous features: growth rate, substrate uptake rate, and the ratios of substrate co-utilizations (glucose, fructose, galactose, gluconate, glutamate, citrate, xylose, succinate, malate, lactate, pyruvate, glycerol, acetate and NaHCO₃, as shown in Figure 1). Since the training features include both categorical and continuous ones, "OneHotEncoder", a function of the Python scikit-learn module, was used to convert categorical feature values into real numbers. Each feature was then standardized into zero mean and unit variance as assumed by many ML approaches. For each predicted flux, we normalized the training dataset via the min-max method into the interval [0, 1]. In addition to the min-max method, we also tested unit-variance-zero-mean standardization for scaling flux values, and the result was quite similar.

6.4.3. Machine learning algorithm selection

The problem of predicting fluxes is formulated as a regression problem in ML, where a computer program learns from existing data to estimate continuous variables. Twenty-nine regressors were trained to predict the 29 fluxes. We tested three widely-applied ML algorithms, including k-nearest neighbors (k-NN), decision tree, and SVM. To ensure a fair comparison, we performed a grid search for the best parameter set of each algorithm. The detailed parameter sets for 29 regression models can be found in the prediction results on our web page (http://130.101.92.205/influx/svr_both_rbf_shuffle.log). The programming language used for this project was Python 2.7; the numpy and scikit-learn modules were utilized for machine learning [31]. Program files for training the models and testing them are wrapped in Supporting Information 1.
6.4.4. Error evaluation and cross validation

To evaluate the quality of the predictive model, we used mean squared error (MSE) and root mean squared error (RMSE). Depending on different evaluation tasks, we may represent RMSE relatively (RRMSE) with respect to the dynamic range of fluxes. Considering the limited number of samples in the current database, we adopt a 10-fold cross validation. An N-fold cross validation works as follows: All samples in our database are spliced into N equal parts. In each iteration, N-1 parts are used as the training set, while the remaining as the test set. In the next iteration, the test set will be rotated to another part of the data, and the training set will consist of all other samples. This procedure will stop when all parts of the data have been incorporated into the test set exactly once, and training set exactly N-1 times. Finally, the accuracy of the model can be calculated by checking the prediction result in each sample.

6.4.5. Stoichiometric constraints and boundary

One unique feature of our method is incorporating the overall mass balance through central metabolic pathways. The stoichiometric equations in Figure 1 under steady state are summarized as follows:
Specifically, $v_1$ represents the flux from carbon substrate (either glucose or galactose) since both glucose and galactose can be catabolized to G6P, $v_{aal}$ and $v_{aal2}$ represents fluxes involved in biomass building block synthesis [32], while vbm represents carbon fluxes going to biomass from different precursors.

A series of linear constraints can be derived from the stoichiometric equations above and used to restrain fluxes predicted by the ML methods:

\[
\begin{align*}
\text{G6P: } v_1 &= v_2 + v_{10} + v_{bm_{g6p}} \\
\text{F6P/FBP: } v_2 + v_{15} + v_{16} + 100 \cdot \text{ratio}_{fructose} &= v_{bm_{f6p}} + v_3 \\
\text{DHAP: } v_3 + 100 \cdot \text{ratio}_{glycerol} &= v_4 \\
\text{GAP: } v_3 + v_4 + v_{14} + v_{15} + v_{23} &= v_5 + v_{16} + v_{bm_{gap}} \\
\text{3PG: } v_5 &= v_6 + v_{bm_{3pg}} \\
\text{PEP: } v_6 &= v_7 + v_{28} + v_{bm_{pep}} \\
\text{PYR: } v_7 + v_{25} + v_{29} + 100 \cdot \text{ratio}_{pyruvate} &= v_8 + v_{27} + v_{bm_{pyr}} \\
\text{AceCoA: } v_9 &= v_{17} + v_{24} + v_{26} + v_{bm_{accoa}} = v_8 \\
\text{Ru5P: } v_{11} &= v_{12} + v_{13} \\
\text{R5P: } v_{13} &= v_{14} + v_{bm_{r5p}} \\
\text{E4P: } v_{15} &= v_{bm_{e4p}} = v_{16} \\
\text{STP: } v_{14} &= v_{16} \\
\text{X5P: } v_{12} + 100 \cdot \text{ratio}_{xylose} &= v_{14} + v_{15} \\
\text{6PG: } v_{10} + 100 \cdot \text{ratio}_{gluconate} &= v_{11} + v_{25} \\
\text{CIT: } v_{17} + 100 \cdot \text{ratio}_{citrate} &= v_{18} \\
\text{ICIT: } v_{18} &= v_{19} + v_{24} \\
\text{AKG: } v_{19} + 100 \cdot \text{ratio}_{glutamate} &= v_{20} + v_{bm_{akg}} \\
\text{SUC: } v_{20} + v_{24} + 100 \cdot \text{ratio}_{succinate} &= v_{21} + v_{oa1} \\
\text{FUM: } v_{21} + v_{oa2} &= v_{22} \\
\text{MAL: } v_{22} + v_{24} + 100 \cdot \text{ratio}_{malate} &= v_{23} + v_{29} \\
\text{OAA: } v_{23} + v_{28} &= v_{17} + v_{bm_{oaa}}
\end{align*}
\]
Among equations listed above, Eq. 22 indicates the case for co-metabolism of both C6 sugars. Meanwhile, a list of inequality constraints can be drawn, given that all biomass fluxes are non-negative:

\[
\begin{align*}
v_1 - v_2 - v_{10} & \geq 0 \\
v_2 - v_3 + v_{15} + v_{16} + 100 \cdot \text{ratio}_{\text{fructose}} & \geq 0 \\
v_3 + v_4 - v_5 + v_{14} + v_{15} - v_{16} + v_{25} & \geq 0 \\
v_5 - v_6 & \geq 0 \\
v_6 - v_7 - v_{28} & \geq 0 \\
v_7 - v_8 + v_{25} - v_{27} + v_{29} + 100 \cdot \text{ratio}_{\text{pyruvate}} & \geq 0 \\
v_8 - v_9 - v_{17} - v_{24} - v_{26} & \geq 0 \\
v_{13} - v_{14} & \geq 0 \\
v_{15} + v_{16} & \geq 0 \\
v_{19} - v_{20} + 100 \cdot \text{ratio}_{\text{glutamate}} & \geq 0 \\
v_{17} + v_{23} + v_{28} & \geq 0 \\
v_{21} + v_{22} & \geq 0
\end{align*}
\]

Among all inequality constraints, constraint (Eq. 39) works well except for the case of zwf knockout, where the directions of Eq. 39 could be reversed [33].

6.4.6. Flux adjustment using stoichiometric constraints
We also adopted a quadratic programming method similar to minimization of metabolic adjustment (MOMA) [34], to adjust fluxes to satisfy the stoichiometric constraints. The CVXOPT package for Python was employed here for quadratic programming [35]. The optimization problem is modeled as

\[
\begin{align*}
\text{Minimize} & \quad f(v) = \sum_{i=1}^{29} (\text{Scaled}(v_i) - \text{Scaled}(\hat{v}_i))^2 \\
\text{Subject to} & \quad S \cdot v = 0, \\
& \quad A \cdot v \geq 0,
\end{align*}
\]

(43)

where the vector $\hat{v} = [\hat{v}_1, \ldots, \hat{v}_{29}]$ is the flux values predicted by ML, the vector $v = [v_1, \ldots, v_{29}]$ is the flux values to be solved in this optimization problem, the function “Scaled (⋅)” uses Min-Max scaling to scale all fluxes into the range [0, 1], the matrix “S” is obtained from all equality constraints from Eq. 22 to Eq. 30, and the matrix A is obtained from all inequality constraints from Eq. 31 to Eq. 42. Scaling fluxes into the same range is done to avoid bias because fluxes have different dynamic ranges. The root mean squared error (RMSE) is used to evaluate the quality of flux prediction by examining the deviation of the predicted flux profile from the $^{13}$C-MFA flux. The objective function $f(v')$ can be rewritten into standard quadratic programming problem using the following steps:

\[
f(v) = \sum_{i=1}^{29} (\text{Scaled}(v_i) - \text{Scaled}(\hat{v}_i))^2 = \sum_{i=1}^{29} \left( \frac{v_i - \text{Min}_i}{\text{Max}_i - \text{Min}_i} - \frac{\hat{v}_i - \text{Min}_i}{\text{Max}_i - \text{Min}_i} \right)^2 \\
= 2 \cdot \sum_{i=1}^{29} \left( \frac{v_i^2}{(\text{Max}_i - \text{Min}_i)^2} + \frac{1}{\text{Max}_i - \text{Min}_i} \cdot \frac{v_i}{\text{Max}_i - \text{Min}_i} + \frac{\hat{v}_i^2}{(\text{Max}_i - \text{Min}_i)^2} \right) \\
\]

(44)
where $Max_i$ and $Min_i$ are the range of the $i^{th}$ flux. Since the last term $\frac{1}{2} \left( \frac{\hat{v}_i}{Max_i - Min_i} \right)^2$ and the coefficient 2 are constants, they can be omitted from the objective function. Hence, Eq. 43 can be rewritten in standard quadratic programming form as

$$\text{Minimize } f(v) = \frac{1}{2} \sum_{i=1}^{29} (v_i)^2 + \sum_{i=1}^{29} \frac{-1 \cdot v_i \cdot \hat{v}_i}{(Max_i - Min_i)^2}$$

Subject to $S \cdot v = 0$, $A \cdot v \geq 0$. (45)

For the upper and lower boundaries of each flux, i.e., $Max_i$ and $Min_i$, we use the maximal and minimal values observed in multiple datasets as the default values (shown in Figure 6.2). Users can manually set desired values for the upper/lower bounds of any specific flux in the MFlux webpage, or they can opt to not use any boundaries. For instance, users can simply set the bound value of a certain flux as zero if this specific gene is knocked out.

6.4.7. Constraint programming and input checking

To ensure user inputs are reasonable, MFlux first checks the satisfiability of input values. This system scans the inputs (e.g., growth rates, oxygen usage, and substrate uptake rates) and determines whether they are biologically meaningful (e.g., unrealistic high cell growth rate). If a set of inputs are suspected to be unreasonable, MFlux reports an error to warn the users.

6.4.8. Overall system design

Different parts of MFlux mentioned above are put together as illustrated in Figure 3. The prediction on 29 fluxes is done via an RBF-kernel SVM, whose outcome will be tuned by constraint programming to generate final prediction. Users can set boundary constraints to
represent information about genes that are knocked out on the species, and such information will be used by constraint programming. If parameter inputs by users are not biologically meaningful, a warning message will be attached in the final result. In the future, users will also have the option to enter fluxes and settings of their own experiment to enrich our database and improve the prediction accuracy of MFlux.

6.5. Results and Discussion

6.5.1. Pathway map and statistical analysis results

The core metabolism of bacteria is summarized into a pathway map in Figure 1. Considering the availability of information, 29 major fluxes with 14 potential substrates were used to represent a universal heterotrophic carbon metabolism for non-photosynthetic prokaryotic species, which includes glycolysis, the tricarboxylic acid (TCA) cycle, the pentose phosphate (PP) pathway, the Entner–Doudoroff (ED) pathway, the glyoxylate shunt, and the anaplerotic pathways. The anaplerotic pathway fluxes cannot be determined when [1-\(^{13}\)C] glucose is fed as a sole labeled substrate [36]. Information on the anaplerotic pathway is either incomplete or not precise in many publications in our database. Consequently, we simplified the anaplerotic pathway into two reversible fluxes. Similarly, we ignored several overflow fluxes which occasionally appear in \(^{13}\)C-MFA of anaerobic metabolisms (e.g., the secretion of formate, butyrate, or pyruvate), because of lacking sufficient samples for efficient machine learning. Omission of those fluxes can also partially explain the high prediction error in specific fluxes (e.g., v8: Pyruvate \(\rightarrow\) Acetyl-CoA).

By statistical analysis, we determined the variation between each flux profile and the average flux profile from our \(^{13}\)C-MFA database. The average value, the range, and the 95%
confidence interval for each individual flux are shown in Figure 2. The most conservative fluxes include the non-oxidative pentose phosphate pathway and the glyoxylate shunt. The former pathway supplies precursors for bio-synthesizing amino acids (i.e., histidine, phenylalanine, tryptophan, and tyrosine) and nucleotides. The latter acts as an alternative carbon-conserving path to the TCA cycle and is inhibited by the presence of glucose (most $^{13}$C-MFA is based on the glucose metabolism). All 29 fluxes were found to have a narrow 95% confidence interval (compared to possible flux ranges), suggesting that fluxes of bacterial species in our database varies in a relatively small range. This is because most $^{13}$C-MFA studies are focusing on models species (e.g., *E. coli* and *B. subtilis*) and glucose-based metabolism, while there are much less MFA efforts to study non-model species or metabolism of carbon substrates other than sugars (i.e., bias of fluxome research across).

6.5.2. Optimization of algorithm and parameters

To decide the most suitable ML algorithm, we first performed a grid search on the parameter space, based on the collection of a wild type (WT) database for initial screening. After one week of running the search program on the server, the best results of the three different algorithms (for SVM, only the linear kernel was considered here) were presented in Figure 4. Evidently, SVM made better predictions than either decision tree or k-NN on most fluxes. After this step, we carried out a second round of grid searching to optimize parameters and improve the performance of SVM on the whole phenotype (WP) database (both WT and engineered). Both the linear kernel and radial bias function (RBF) kernel were included in this round of searching.

Better cross-validation results were expected from the SVM model trained on the WT database, rather than on the WP database, considering that sophisticated genetic variations are
not included in the WT database. However, cross-validation results refuted our initial thought: the models from the WP database demonstrated better performance than those trained on the WT database (Figure 5). This result can be interpreted as meaning that the size of the training dataset is a major factor in determining the model quality, especially when the training database size is relatively small (the size of WT dataset is 154, and the size of WP dataset is 450). We also compared the SVM results with the linear kernel and the RBF kernel, and the RBF kernel showed slightly better performance (Figure 6). We finalized the parameter setting of MFlux by taking the parameter set which output the best cross-validation result. For all the algorithms tested, v11 (the second step of the oxidative PP pathway) and v24 (the glyoxylate shunt) are insensitive in terms of RRMSE. Two factors may contribute to this problem: v11 and v24 have relatively narrow numerical ranges, and consequently even small numerical variations will generate larger relative errors for both fluxes. Meanwhile, genetic modifications may influence both v11 (e.g., zwf knockout [37]) and v24 (e.g., ppc knockout [38,39]). For instance, knocking out zwf in E. coli will cause a zero flux in v10 (the oxidative pentose phosphate pathway, OPP pathway). However, lack of sufficient information on flux re-organization mechanisms in engineered microbes reduced ML predictability. This is because most engineered microbial fluxomics studies are focused on a few model species such as E. coli. To resolve this problem, the MFlux platform allows users to manually set the boundaries of central fluxes to improve prediction quality (e.g., give a zero flux through the OPP pathway for E. coli zwf mutant).

6.5.3. Flux correction by quadratic programming

After parameter optimization, the SVM models equipped with the best parameter sets can predict with relatively small variation. However, the flux profile predicted by the ML method
does not necessarily satisfy the inherent stoichiometric constraints of metabolic networks because the ML method does not consider them. Sometimes the situation could get even worse: Specific fluxes predicted by the ML algorithm may go beyond a reasonable range (e.g., the predicted glyoxylate shunt may have a negative value). To address those issues, we employed quadratic programming for flux correction, as described in the Methods section. More rational results with improved accuracy are expected after flux correction. An essential assumption of this step is that ML predictions are relatively close to the real values reported in the literature. This hypothesis is backed up by our cross-validation results and further validated in the following case studies. Notably, biomass equations may have differences among MFA papers (e.g., equation 18 and 19). Considering the variations in biomass fluxes, the revised quadratic programming didn’t include constraints from succinate mass balance equation (i.e., succinyl-CoA flux towards biomass synthesis).

6.5.4. MFlux case studies

To demonstrate the functionality of MFlux, we carried out tests on twenty cases, and the results are illustrated in Figure 7. General information for each case is listed in Table 1, and comprehensive results are included in Supporting Information 2. In general, MFlux can achieve decent flux predictions. Here we will demonstrate two cases which are case 8 and 16 in Supporting Information 2. In case 8, *B. subtilis* strain uptakes the mixed substrates succinate and glutamate.

To illustrate mixed substrates co-metabolisms, we tested MFlux with $^{13}$C-MFA data of *B. subtilis* strain reported by Chubukov *et al.* [40]. Microbial fermentation fed with multiple substrates of low price is promising for the biotechnology industry. However, there are few quantitative analyses of this topic. In this test, we adopted the same set of parameters found in
the literature (Supporting Information 2, Case 8) as the inputs of MFlux. For flux correction, we
directly took the default boundary setting for quadratic programming. A comparison of flux
profiles reported by $^{13}$C-MFA, predicted by ML, and predicted by MFlux is illustrated in Figure
8. ML and MFlux produce good predictions on most fluxes, closely matching the $^{13}$C-MFA flux
profile (the RMSE is less than 5). For ML, the predictions have large variation on specific fluxes
(e.g., v11 - oxidative PP pathway and v19 – TCA cycle). Quadratic programming can further
adjust flux profiles and reduce deviations of flux predictions. The corrected flux profiles also
meet the basic stoichiometric relationship of the metabolic network. The final prediction from
MFlux shows improvement, with RMSE reduced to 3.2.

In case 16, *G. thermoglucosidasius* grows under microaerobic conditions. *G.
thermoglucosidasius* is a thermophilic and ethanol-tolerant bacterium which can convert both
hexose and pentose into ethanol [30]. To predict its central fluxomes, the parameter set we used
is listed in Supporting Information 2 (with the default boundary setting for flux correction). A
heat map compares $^{13}$C-MFA fluxes with ML-only fluxes and MFlux results (Figure 9). The
results are encouraging: ML alone gives an RMSE of 4.0, while MFlux uses both ML and
quadratic programming to improve the prediction to an RMSE of only 3.0. According to 20 case
studies, the average flux set has very large variations (average RMSE of 33.5) from actual $^{13}$C-
MFA fluxes (Supporting Information 2). In this case, MFlux reduces the deviations of predicted
fluxes from $^{13}$C-MFA values.

For species with genetic modifications in major pathways (cases 2, 3, 4, 12, and 13, *E.
coli* and *C. glutamicum*), MFlux predictions have an average RMSE between 5 and 10, higher
than the RMSE for prediction of wild type strains. Since MFlux is currently unable to capture
complex regulatory mechanisms of flux reorganization, Human-Computer Interaction can be
employed by manually tuning boundary values of certain fluxes to improve flux prediction quality. For example, knocking out \textit{ppc} on \textit{E. coli} may activate the glyoxylate shunt [38,39], so users can assign a non-zero lower boundary of the glyoxylate shunt when running MFlux.

6.5.5. \textbf{Compare flux balance analysis with MFlux for \textit{E. coli} metabolism}

Stoichiometry-based flux balance analysis (FBA) is an important tool to predict unknown cell metabolism. Accurate FBA prediction relies highly on appropriate setting the objective function and the flux constraints appropriately (based on thermodynamics or experimental analysis) [41]. Here, we compare FBA with MFlux for predicting \textit{E. coli} metabolisms. The latest version of \textit{E. coli} iJO1366 genome-scale model (2583 fluxes) was used [42,43]. Two comparative case studies were performed on \textit{E. coli} fluxomes: One case for glucose based $^{13}$C-MFA via parallel labeling experiments [12], the other case for glucose and glycerol co-utilization (unpublished data from the Shimizu Group). Neither of the test cases was included in the training database of MFlux. Given $^{13}$C-MFA results as the control, MFlux results have smaller RMSEs than FBA predictions. In the first case, the FBA has an RMSE of 11.3, while MFlux has an RMSE of 6.5 (Figure 10a). In the second case, the FBA has an RMSE of 22.5, while MFlux has an RMSE of 5.1 (Figure 10b). To circumvent variations caused by alternative solutions in FBA, we also employed pFBA and geometricFBA in cases study [44,45] (results were included in Supporting Information 3). In general, pFBA didn’t show better results compared with FBA for either case, while geometricFBA did not converge during our calculation.

FBA alone has been shown to give good predictions of growth rate as well as input and output fluxes, but not of intercellular fluxes [2,46]. It is difficult to obtain actual P/O ratios, the non-growth associated maintenance energy, the oxygen flux, and the transhydrogenase activities [47]. These energy/cofactor variables strongly affect the fluxes in the oxidative PP pathway.
(NADPH generation) and the TCA cycle (NADH, NADPH, and FADH$_2$ generation). Without proper flux constraints and objective functions, it is challenging for FBA to narrowly determine intracellular fluxomes in suboptimal metabolisms, especially for co-metabolism dual substrates (i.e., there are large solution spaces for the cell metabolism to optimize biomass growth using two substrates). As a complementary tool, MFlux may offer a quick metabolic overview and provide reasonable flux boundaries to reduce FBA solution spaces when proper constraints for FBA are not available.

6.5.6. Perspective of metabolic robustness and machine learning of fluxome patterns

‘Robustness’ was originally defined as closed-loop process stability under perturbations in the control field. This definition is applicable to biochemical networks. To maintain the physiological output (i.e., the fluxome) within a desired range, microorganisms employ sophisticated control disciplines at different architecture levels, from the genome to the phenotype [48]. In contrast to chaotic transcriptional profiles, the microbial fluxome shows robustness so that cells can survive in constantly-altering environments or in response to genetic mutations [49,50,51]. Metabolic rigidity at the flux level was first reported by Stephanopoulos in the early 1990s [52,53]: NADPH is important for anabolism in the exponential growth phase, and the flux ratio around glucose-6-phosphate is rigid to form NADPH at the oxidative PP pathway [53]. Moreover, 12 precursors from the central metabolism are required for biomass formation, which all have relatively small variations (mainly dependent on biomass compositions). Due to both thermodynamic and mass balance constraints, cell metabolism aims to minimize variations in flux ratios under environmental perturbations. This rule also works for engineered microbes with moderately over-expressed pathways or strains from random
mutations. Those metabolic patterns can be identified by computational intelligence methods to facilitate fluxome prediction.

Flux pattern recognition enables MFlux to predict metabolism of new species by learning from a small set of fluxome information from the same genus. For example, the metabolisms of *Pseudomonas aeruginosa*, *Pseudomonas fluorescens*, and *Pseudomonas putida* have been studied by $^{13}$C-MFA in the past decade[54,55,56,57,58]. The results show that different *Pseudomonas* species employ remarkably identical fluxomics types: They employ a highly active ED pathway for glycolytic metabolism and keep a low flux on the PP pathway for biomass synthesis, due to the lack of the *pfk* gene [59]. The ED pathway has less cost for protein formation than the Embden–Meyerhof–Parnas (EMP) pathway, yet only one ATP is formed per glucose [60,61]. *Pseudomonas* species have slow cell growth rates, and their aerobic metabolisms do not yield any by-products. They also demonstrate a very active pyruvate shunt (malate $\rightarrow$ pyruvate) and NADPH overproduction flux (a benefit for counteracting oxidative stress). On the other hand, the TCA cycle in *Pseudomonas* species shows plasticity under genetic and environmental variations [62], and can respond to increased ATP and NADH demands under stress conditions [63].

For different bacterial families (e.g., *E. coli* and *Bacillus*), their fluxomes (e.g., glucose metabolisms) can also be similar, because central fluxes in catabolism are regulated by energy and building block requirements that show much smaller variations than genome or transcriptional differences. On the other hand, change of carbon substrates may alternate flux distributions. For example, co-utilization of glucose and glycerol (case study 3) in *E. coli* cause significant re-organization of fluxomes. In a same microbial strain, different fluxome patterns can be employed for metabolizing different substrates (e.g., glucose based fluxome vs acetate
based fluxomes). Recognizing these metabolic patterns allows the use of a relatively small training database to perform a decent metabolic prediction of diverse metabolic types. Consequently, these common principles of certain classes of microorganisms can be captured by machine learning for fluxome predictions.

6.5.7. Limitations of machine learning

There are still several major challenges regarding MFlux. First, the $^{13}$C-MFA flux in literature database may have errors and bias, which would be included in the learning/training process of MFlux and lead to further variations. For example, current $^{13}$C-MFA studies are not evenly distributed among a broad scope of microbial genus. Most reported MFAs are concentrated in a few model microbial species using glucose as substrates, while there are much fewer papers on non-model species or metabolism of diverse substrates other than sugar. Such problem (bias of fluxome in the database) can be resolved after more papers on $^{13}$C-MFA can be published for non-model species.

Second, the predictability of ML is limited to species and pathways that are already included in learning. More information and effort are required to deal with cases of strains with engineered pathways that hijack flux for synthesis of diverse commodity chemicals [13]. In future versions of MFlux, new metabolic knowledge and rules should be applied for flux corrections.

Third, it is still difficult to incorporate regulation mechanisms into the current model due to insufficient $^{13}$C-MFA studies. For instance, various catabolite repression mechanisms regulate the cell fluxome in the presence of multiple substrates (e.g., glucose shows catabolite repression for fast growing E. coli when both glucose and glycerol are available, Figure 10) [64]. These
hierarchy regulations among substrate utilization can be dependent on growth rates or can differ among microbial species (E. coli, Bacillus, and Corynebacterium).

Fourth, when oxygen is not available, fast sugar utilization will activate mixed acid fermentation (e.g., by utilizing lactate dehydrogenase and pyruvate formate lyase) to produce complicated overflow metabolites. This mechanism is also furnished in yeast and mammalian cells. However, 13C-MFA studies on anaerobic metabolisms are much less frequent than on aerobic metabolisms. MFlux cannot predict the complicated patterns of overflow fluxes at this stage.

Lastly, ML cannot directly estimate fluxes for carbon sources which are not part of the learning dataset. To predict fluxomes for new substrates, users need to make assumption that similar entry-points of carbon sources into the central metabolic network may cause similar flux distributions (e.g., sucrose has to be treated as a combination of glucose and fructose).

6.6. Conclusion

This proof-of-principle study demonstrates that AI methods can facilitate fluxomics research with reasonable precision. 13C-MFA is a very small field: There are just hundreds of MFA research papers on microbial species published in the past two decades. In the long term, ML methods may solve this problem: With a large and reliable fluxomics dataset and more information from 13C-MFA and AI scientists, the future model can make broad-scope metabolism predictions. To sum up, MFlux presents the first platform that incorporates machine learning, constraint programming, and quadratic programming in the field of fluxomics. It will inspire the development of similar computational tools to advance omics and metabolic engineering fields [47,65].
6.7. Supporting information

Appendix II S1 MFlux Computer Program (Source codes).

Appendix II S2 Results of 20 case studies: Detailed information for 20 cases studies using MFlux, including literature references, input conditions, $^{13}$C-MFA fluxes, the flux profiles predicted by only Machine Learning, and the flux profiles predicted by MFlux with additional constraints.

6.8. References


36. Zhao J, Baba T, Mori H, Shimizu K (2004) Global metabolic response of *Escherichia coli* to *gnd* or *zwf* gene-knockout, based on $^{13}$C-labeling experiments and the measurement of enzyme activities. Applied Microbiology And Biotechnology 64: 91-98.


*denitrificans* using $^{13}$C-labeled glucose. Journal of the Taiwan Institute of Chemical Engineers 43: 181-187.


Table 6.1. Summary of 20 cases of study

<table>
<thead>
<tr>
<th>Species</th>
<th>Carbon source</th>
<th>Oxygen condition</th>
<th>Reactor</th>
<th>Genetics</th>
<th>Case</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>E. coli</em></td>
<td>Glc</td>
<td>aerobic</td>
<td>shake tube</td>
<td>WT</td>
<td>1</td>
<td>(Crown et al. 2015)</td>
</tr>
<tr>
<td><em>E. coli</em></td>
<td>Glc</td>
<td>aerobic</td>
<td>shake flask</td>
<td>ppc KO</td>
<td>2 - 4</td>
<td>(Fong et al. 2006)</td>
</tr>
<tr>
<td><em>B. subtilis</em></td>
<td>Glc</td>
<td>aerobic</td>
<td>shake flask, CSTR</td>
<td>WT, spo0A KO</td>
<td>5 - 7</td>
<td>(Tannler et al. 2008)</td>
</tr>
<tr>
<td><em>B. subtilis</em></td>
<td>Multiple substrates</td>
<td>aerobic</td>
<td>shake flask</td>
<td>mutant</td>
<td>8 -11</td>
<td>(Chubukov et al. 2013)</td>
</tr>
<tr>
<td><em>C. glutamicum</em></td>
<td>Glc</td>
<td>aerobic</td>
<td>shake flask</td>
<td>WT</td>
<td>12</td>
<td>(van Ooyen et al. 2012)</td>
</tr>
<tr>
<td><em>C. glutamicum</em></td>
<td>Glc</td>
<td>aerobic</td>
<td>shake flask</td>
<td>mutant</td>
<td>13</td>
<td>(Bommareddy et al. 2014)</td>
</tr>
<tr>
<td><em>P. denitrificans</em></td>
<td>Glc</td>
<td>aerobic, microaerobic</td>
<td>fermentor</td>
<td>WT</td>
<td>14, 15</td>
<td>(Wang et al. 2012)</td>
</tr>
<tr>
<td><em>G. thermoglucosidasius</em></td>
<td>Glc</td>
<td>microaerobic</td>
<td>shake flask</td>
<td>WT</td>
<td>16</td>
<td>(Tang et al. 2009c)</td>
</tr>
<tr>
<td><em>Thermoanaerobacter sp.</em></td>
<td>Xyl</td>
<td>anaerobic</td>
<td>Sealed bottle</td>
<td>WT</td>
<td>17, 18</td>
<td>(Hemme et al. 2011)</td>
</tr>
<tr>
<td><em>D. vulgaris</em></td>
<td>Lac</td>
<td>anaerobic</td>
<td>Sealed bottle</td>
<td>WT</td>
<td>19</td>
<td>(Tang et al. 2007b)</td>
</tr>
<tr>
<td><em>G. metallireducens</em></td>
<td>Ace</td>
<td>anaerobic</td>
<td>Sealed bottle</td>
<td>WT</td>
<td>20</td>
<td>(Tang et al. 2007a)</td>
</tr>
</tbody>
</table>

Table notes: Glc: glucose, Xyl: xylose, Lac: lactate, Ace: acetate, KO: knockout
Figure 6.1: A universal central metabolic pathway for bacteria: The central carbon metabolic pathway is simplified into 29 fluxes in MFlux.
Figure 6.2. Statistical analysis of central metabolic fluxes collected in our database. “Flux range” represents variations of each fluxes among $^{13}$C-MFA database; “95% confidence interval” represents 95% of flux data were within a small range; “Average flux value” are the mean of flux values from $^{13}$C-MFA database.
Figure 6.3. Flow chart of MFlux algorithm. This diagram is to illustrate the detailed procedures for our algorithm.
Figure 6.4. A comparison of three different algorithms: SVM, kNN, and decision tree: The best cross-validation results on 29 fluxes are compared. All tests in this step were performed only on the WT database.
Figure 6.5. Best results by SVM for WT and WP databases. Both the linear and the RBF kernels are considered in a grid search, and the results from WP database is much better than from the WT database.
Figure 6.6. A comparison between the linear kernel and the RBF kernel for SVM. The results are quite similar.
Figure 6.7. Summary of root mean squared error (RMSE) from 20 case studies: averaged flux from $^{13}$C-MFA database; machine learning, and MFlux. The average RMSE is 7.7 from machine learning alone and 5.6 from MFlux. The RMSE is calculated by:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{29} (v_i - \hat{v}_i)^2}{29}}$$
Figure 6.8. A comparison of $^{13}$C-MFA, the flux predicted by ML, and the flux predicted by MFlux in case 8. *B. subtilis* was incubated in a shake flask (37 °C, 300 rpm, aerobic condition), and supplied with labeled succinate and glutamate as carbon sources in M9 minimal medium. The detailed information is in supporting file 2.
Figure 6.9. A comparison of $^{13}$C-MFA, the flux predicted by ML flux, and the flux predicted by MFlux. *G. thermoglucosidasius* M10EXG was incubated in sealed bottles (micro-aerobic condition), supplied with glucose as a carbon source. RMSE$_{ML} = 4.0$, RMSE$_{MFlux} = 3.0$. The detailed information is in supporting file 2.
Figure 6.10. A comparison of $^{13}$C-MFA, MFlux and the flux predicted by FBA. FBA Analysis is simulated by E. coli iJO1366 model with defaulted boundary settings from the reference (Orth et al. 2011). (A) E.coli fluxome of glucose metabolism was precisely measured via parallel labeling experiments (a recent paper not in our database) (Crown et al. 2015). RMSE$_{FBA}$ = 11.3, RMSE$_{MFlux}$ = 6.5. (B) E. coli fluxome of glycerol and glucose co-metabolism were measured by Dr. Yao and Dr. Shimizu (unpublished data). E. coli strain was cultured in chemostat fermentor with a working volume of 1 L (37°C). The dilution rates in the continuous culture were 0.35 h$^{-1}$. [1-$^{13}$C] glucose and [1,3-$^{13}$C] glycerol were used for tracer experiments. The flux calculation is based on previous method (Fong et al. 2006; Peng et al. 2004). RMSE$_{FBA}$ = 22.5, RMSE$_{MFlux}$ = 5.1
CHAPTER SEVEN
ENABLE FAST LITERATURE ANALYSIS BY TEXT MINING AND BIG DATA TECHNOLOGY

7.1. Abstract
Information acquisition by human being is severely limited by the speed and time of reading, as well as the background of readers. To gain a deep understanding on a specific research subject requires even longer time of training and learning. With information gradually digitalized, text mining method provides an automatic way for literature analysis. In this study, we built up a workflow integrating text mining and the emerging Big Data technology for fast literature analysis. We also performed several case studies to demonstrate its functionality. A comparison between ‘metabolic engineering’ and ‘synthetic biology’ finds that energy metabolism terms (i.e., ‘NADH’, ‘NADPH’, ‘ATP’) and non-glucose carbon substrate (i.e., ‘xylose’, ‘acetate’, ‘glycerol’) have significant higher frequency for ‘metabolic engineering’, while researchers on ‘synthetic biology’ talk more on regulatory modules such as ‘circuit’, ‘loop’, ‘switch’, ‘IPTG’, ‘LacI’. In another case the transition of ‘metabolic engineering’ between ‘2000 - 2009’ and ‘2010 - 2015’ has also been identified: more focuses were put on model species such as ‘B. subtilis’ and ‘P. pastoris’ as microbial cell factories and ‘hydrogen’ during 2000 ~ 2009, while the focuses have been shifted to photosynthetic species ‘cyanobacteria’ as well as ‘butanol’, ‘isobutanol’, ‘lipid’. Each comparison takes less than three minutes and is easily extended with various specific searching settings. To sum up, this proof-of-principle study demonstrates that Big Data technique can quickly capture similarity/difference and provide quantitative information,
which enable us have more reflections over the past and make more reasonable choices in the future.

7.2. Introduction

As a cornerstone of human society, technology development is accompanied with emerging fields and changing focuses in different field. Understanding those similarities and differences between various fields, as well as the same field of different time periods will not only bring novel insights to researchers, but also provide invaluable historical experiences for a broader audience (e.g., industry, government). To gain a deep understanding over a specific subject, information from Wikipedia is far away from sufficient; extensive reading over thousands of papers is a perquisite, which is very time-consuming yet difficult to avoid bias.

The trend of information digitalization was emerged with wide adoptions of personal computer and Internet. (Arms 2000) It has revolutionized the manner of information record and storage in human civilization: tons of information can be stored in small hard disks and can be easily duplicated and spread. Based on the increasing availability of digital information, text mining provides an automatic approach for information analysis (Dorre et al. 1999). Text mining have been widely adopted in many fields such as biomedical literature analysis (Cohen and Hersh 2005), systems biology (Ananiadou et al. 2010), and human phenome (van Driel et al. 2006).

As the size of data and information is increasing with time, the concept of ‘Big Data’ also comes out several years ago. Correspondingly, analyzing large amount of data comes across the limitation of hardware. For instance, searching a specific term from 60 GB of data in txt formate
takes about 10 hours in a powerful working station, which is unacceptable if similar search actions are carried out all the time. Two major approaches have been developed by Google, to deal with those challenges: the first one is MapReduce, which is suitable for batch processing of large datasets; the other one is BigQuery, which works well with interactive analysis over large datasets (Tigani and Naidu 2014).

In this study, we propose to build up a workflow which provides rapid literature analysis based on an integrated platform of text mining and BigQuery. Development of such a workflow will bring a novel approach for fast acquisition of professional knowledge.

7.3. Methods

7.3.1. Database availability and record structure

All full-text papers are downloaded from NCBI PMC database (http://www.ncbi.nlm.nih.gov/pmc/tools/ftp/). PMC database contains ~1.1 million full-text papers (without images) from ~5600 different journals with a focus on the biomedical research. This database is updated weekly; hence, the total number of papers and journals are still increasing (shown in Figure 7.1.).

There are two basic formats of papers in PMC database: .txt type and .nxml type. Txt files have to lose lots of essential contents of original papers, due to their format limitation. In contrast, nxml files contain most important information of papers except images. Therefore, we choose nxml files for further literature analysis.
Nxml files contain structured information extracted from papers. In general, there are three parts for each nxml file as shown in Figure 7.2:

1. Front: which contains information such as paper title, pmcid, author name, author email, and author affiliation.

2. Body: the major part of the manuscript, including the section of methods, results, discussions, supporting information.

3. Back: list of references

Most information illustrating research topics of papers normally is included in the part of ‘body’.

7.3.2. Text mining methods

To extract information from each paper (nxml file), a powerful language processing toolbox is indispensable. The most common Natural Language Processing libraries includes NLTK (in Python) (Bird 2006; Bird et al. 2009), Stanford NLP Toolbox (in Java), OpenNLP (in Java), Gate (in Java). Different libraries have their advantages and disadvantages; for nxml format files, Beautiful Soup library (in Python) can work well with them. Therefore, we employed Beautiful Soup and NLTK library together in this study. All programming language is Python 2.7.

To demonstrate the functionality of our workflow, we only extract PMC id, year, whole body, unique words in body, and frequency of each word in this work. A simple procedure of text mining is listed below:

1. A paper in nxml format is read in by Beautiful Soup library

2. The PMC id, article name, as well as the body, are extracted by using Beautiful Soup library with following codes:

   \[
   \text{pmc\_id } = \text{soup.find\_all('article-id',attrs={"pub-id-type": "pmc"})}
   \]
To our surprise, the part of body extracted by Beautiful Soup function still contains the front and the back part. Hence, we artificially remove strings of both parts by using ‘replace’ function.

3. We remove all symbols in the body and tokenize body text by the following code:

```python
real_text = ((((((real_body.replace("\n"," ")).replace(',',' ').replace('.',' ').replace('(',' ').replace(')',' ').replace(':',' ').replace(';',' '))).replace(':',' ')).replace(';',' '));
paper_word = nltk.word_tokenize(real_text);
```

4. We employ the library of common English words (stopwords) such as ‘I’, ‘and’, and add a few more. The final library of common words is about 200. Since those words do not have any exact meanings related with research topics, we can remove them from the text. The total size of text strings can also be reduced to about 55% of their original size through step 2 - 4.

5. We convert all words to be lowercase, and then extract all unique words and their respective relative frequency. Considering of significant difference in text size of different papers (the average word number for papers is about 5,800; a technical note can be as short as 1,500 words; while a reviewer paper can be as along as 14,000 words), to equalize the impact of each paper, we use relative frequency rather than absolute frequency in our analysis.

6. We store the following information in a CSV file with five columns:

   PMC id, year, body, unique word, relative word frequency

7.3.3. Fast search via BigQuery
The final CSV file generated from text mining is around 21.7 GB, about 30% volume of their original files. This file contains more than 1 million records, searching a single term takes at least several hours using traditional methods. To accelerate this process, we employed Google BigQuery for the search process. BigQuery is the commercialized product of Dremel, the search engine for Google insiders. (Tigani and Naidu 2014) Equipped with columnar structure data, BigQuery performs a searching task in parallel (on several thousands of servers), and can finish a searching task over several billion of records within ten seconds.

First, we need create a bucket in Google Cloud and upload the large CSV file into the bucket. A stable internet connection is necessary because the uploading process may take several hours.

Second, we create a project under BigQuery and create a Big Table under this project. During table creation process, we need to get the link address of the big CSV file in Google Cloud and use for data uploading. Also, we need to define the names of columns, as well as their type and mode in the step of ‘Edit Schema’.

Third, to enable script based Google BigQuery search, we need to download a secret key ‘client_secret. json’ from Google Big Query. This file stores a personalized keyword linked with your Google client account. Thus, each BigQuery task can directly charge your account via this information. After that, we can download the demo code program (in Python) from BigQuery and perform the test in local PC. Once client verification is finished in local PC, we can modify the searching program to perform any search tasks.

7.3.4. Data analysis

For a specific term, we need to input its lowercase string in Python program for BigQuery search. Information of any records contained the specific term will be recorded locally upon searching
request. This local record can be processed through Python program and converted into cumulative relative frequencies of unique words. Through simple sorting over cumulative relative frequencies, word list with a descending order is the output. For more specified search, for instance, search ‘metabolic engineering’ during 2010 ~ 2015, we can modify the program and include more searching parameters in BigQuery.

To visualize top 300 related words of a specific searching term, we employ the library of word cloud in R to display them. For simple quantification, we define the word of highest cumulative relative frequency as 100. Frequencies of other words are normalized in a scale of 100.

To identify the similarity and the difference between two different search term, we perform a simple match between top 500 words for different terms, to determine the percentage of similarity and difference. We also extract those words of difference as an output.

7.4. Results and Discussions

To demonstrate the functionality of our workflow, we carried out several case studies and put the results as below.


The first function of this Big Data workflow is to define related words of a specific term. To run this function, we first search this term within the column ‘body’ via BigQuery. After word list of top cumulative frequency is given, we use R to display their word cloud.
We performed case studies on several terms including ‘metabolic engineering’, ‘environmental engineering’, ‘synthetic biology’, ‘systems biology’, and ‘metabolic flux’, and the results of their word cloud are listed as Figure 7.3. – 7.7. The results provide us much meaningful information:

For instance, the word with highest frequency related with ‘metabolic engineering’ is ‘production’, which reflects the aim of ‘metabolic engineering’ field is to develop production processes through metabolism of microbial cell factories (Stephanopoulous 1999).

In another instance, the most frequent words related with ‘environmental engineering’ is ‘health’, which reveals the motivation of researches on environmental engineering is the health of human beings.

7.4.2. Compare the difference and similarity between two different terms

The second function we want to demonstrate is the comparison of two different terms. This time, we take ‘metabolic engineering’ and ‘synthetic biology’ as a case study, because researchers have different opinions over those two concepts (Nielsen and Keasling 2011; Carothers et al. 2009; Lee et al. 2008; Church et al. 2014). Through this Big Data workflow, we can provide a comparative analysis based on published papers in PMC database.

The result is shown in Table 7.1.: they have a similarity of 69.6% -- indicating that both terms have a large range of scope overlapped, such as ‘gene’, ‘PCR’, ‘E. coli’. The difference between ‘metabolic engineering’ and ‘synthetic biology’ is also apparent: The word of highest frequency for ‘metabolic engineering’ is ‘production’; and there are several words with significantly higher cumulative frequency related with ‘metabolic engineering’, including FBA, NADH, NADPH, ATP, xylose, acetate, glycerol, HPLC, transport, and tolerance. For ‘synthetic
biology”, the most frequent word is ‘gene’, words with significantly higher cumulative frequency are: circuit, loop, IPTG, lacI, RBS, Switch egfp, Phage, and virus. To explain this difference, we can refer to their background and origins: ‘metabolic engineering’ comes from biochemical engineering at the early 1990s (Bailey 1991). With the development of genetic modification tools (e.g., PCR, restriction enzymes), as well as successful commercialization of heterogeneous protein expression (e.g., insulin by Genentech), researchers tried to expand the product scope of biochemical engineering through extensive genetic modifications over microbial metabolism. Thereby, metabolic engineering focus more on energy metabolism (NADH, NADPH, and ATP), substrate utilization (xylose, acetate, and glycerol), engineering and model (transport and FBA), product measurement (HPLC), and microbial physiology (tolerance) (Stephanopoulos et al. 1998b). In contrast, ‘synthetic biology’ first appeared in early the 2000s, coming from the background of biophysics and electric engineering. Researchers tried to redefine biological modules via the standards and rules applied in electrical circuit and chips. Successful demonstration of simple logic parts such as ‘toggle switch’ and ‘oscillator’ in biological systems motivated further work in a similar manner (Gardner et al. 2000; Elowitz and Leibler 2000). With a broad spread of iGEM (International Genetically Engineered Machine) (Smolke 2009; Brown 2007), ‘synthetic biology’ has been widely recognized. And it focuses more on electrical engineering concepts (circuit, switch, and loop), and genetic demonstration tools (IPTG, lacI, RBS, egfp, Phage, and virus)(Canton et al. 2008).

7.4.3. Identify the developing trend of a specific term

Another function we want to show is the comparison with a time specification. This function is quite similar to the second function, except that we need to specify a period during BigQuery
search. We perform two case studies here; one is ‘metabolic engineering’, and the other is ‘biofuel’, and the results are presented in Table 7.2 and 7.3.

Case study on ‘metabolic engineering’:

From Table 7.2., we can see two obvious trends in metabolic engineering from 2000~ to 2010~. The first trend is that the focus of microbial working horse has been changed to from ‘B. subtilis’ and ‘P. Pastoris’ to ‘cyanobacteria’ and ‘C. glutamicum’: B. subtilis is a gram-positive model species widely used to produce vitamin and enzymes, however, the disadvantages for B. subtilis as a host include high maintenance energy (Tannler et al. 2008), existence of many proteases (Zhang et al. 2005). P. Pastoris is a methylotrophic yeast species widely used for recombinant protein expression in industry because of its powerful secretion system and ease to reach high cell density and to avoid contamination in large-scale fermentation (with methanol) (Cregg et al. 2009). However, the disadvantage for P. Pastoris also lies in its methylotrophic property: methanol is volatile and highly toxic; it is highly risky for students with little experience to work on this strain. Researchers turned to species ease to manipulate such as cyanobacteria and C. glutamicum during recent years. The second trend is that the list of hot products has been slightly changed: publications on hydrogen, PHB, and lysine go down, and lipid, butanol, and isobutanol gain more focus. Biosynthesis of PHB and lysine has been successfully commercialized, and most research work supported by the industrial funding will not send for publication. The bottleneck of hydrogen is storage, rather than synthesis. Novel biofuels (butanol, isobutanol, and lipid) with higher energy density than ethanol and ease-to-storage and utilization have gained attentions from both scientific and industrial fields during recent years. With funding pouring into those topics, the outcomes – the number of related papers increased.
Case study on ‘biofuel’:

The development trend of ‘biofuel’ is fascinating, the trend have changed significantly from ‘Lignocellulosic’, ‘hemicellulose’, ‘Emissions’, ‘policy’, ‘Economic’, ‘market’, ‘Feedstock’ from 2000 to 2009, to ‘Glycerol’, ‘acetate’, ‘xylan’, ‘Algae’, ‘microalgae’, ‘Cyanobacteria’, ‘Lipid’, ‘Chromosome’, ‘cDNA’ within recent years. Lots of researches were focusing on degradation and pretreatment of cellulosic material (e.g., lignocelluloses, hemicellulose) to sugar during 2000 to 2009, also there are lots of comments and perspectives on economy/market analysis, and carbon neutral economy. With the support of DOE (Department of Energy) on three energy centers (2007-currently), as well as production of biofuels through metabolic engineering (Atsumi et al. 2008a; Atsumi et al. 2009; Steen et al. 2010), the trend has been shifted to microbial substrates utilization (Glycerol, acetate, and xylan), and photosynthetic hosts (microalgae and cyanobacteria).

7.4.4. Advantages and Limitations of Big Data workflow

Current workflow provides fast, reproducible, and quantitive analysis over literature database. The whole process takes less than 3 min from search to the comparison, and can be easily modified and extended with the addition of more sophisticated functions through programming. For instance, we can easily extract the author information or the institute information in data mining process and store in Big Table. Moreover, such information can be used to track publication records related with specific authors or specific institutes.

The major limitation of the current workflow is the information available in the database. Although PMC is the largest full-text database online, the papers it includes are only 1.1 million currently. Considering the total number of papers available, which counts over 65 million
currently and this number is increasing now. The large gap between the resource we can access and the total number of papers available can be explained by two reasons: first, lots of old papers are still not digitized. Lots of efforts are needed to put all those information into the digital library; second, accessing lots of digital libraries are charged (Hull et al. 2008). There will be copyright issues, as well as conflict of interests to make digital libraries freely access now. Further, data storage and Big Query search will lead to some cost, but is relatively low ($5/TB for either storage or query), which will not be a big issue for further development.

7.5. Conclusion

We have successfully built up a literature analysis workflow based on text mining and Big Data technology. The capability of this workflow has been demonstrated through case studies and can be further enhanced by integrating with other information sources. Considering of its fast speed, reproducible results, scalability with other databases, and ease-to-modify, we believe the further development of this platform will provide deeper insights into literature as well as bringing more benefits for researchers.

7.6. References


### Table 7.1. Comparison of ‘metabolic engineering’ and ‘synthetic biology’, similarity 69.6%

<table>
<thead>
<tr>
<th>Terms</th>
<th>Metabolic engineering</th>
<th>Synthetic biology</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Most frequent words</strong></td>
<td>Production</td>
<td>Gene</td>
</tr>
<tr>
<td><strong>Major differences in 500 most frequent words</strong></td>
<td>FBA (flux model)</td>
<td>circuit/loop</td>
</tr>
<tr>
<td></td>
<td>NADH/NADPH (cofactor)</td>
<td>IPTG/lacI (inducer)</td>
</tr>
<tr>
<td></td>
<td>ATP (energy)</td>
<td>RBS</td>
</tr>
<tr>
<td></td>
<td>Xylose/acetate/glycerol</td>
<td>Switch (regulation)</td>
</tr>
<tr>
<td></td>
<td>HPLC</td>
<td>egfp</td>
</tr>
<tr>
<td></td>
<td>transport/tolerance</td>
<td>Phage/virus</td>
</tr>
<tr>
<td>Terms</td>
<td>Metabolic engineering</td>
<td></td>
</tr>
<tr>
<td>-------------------------------</td>
<td>-----------------------</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2000 ~</td>
<td>2010 ~</td>
</tr>
<tr>
<td><strong>Most frequent words</strong></td>
<td>Metabolic</td>
<td>Production</td>
</tr>
<tr>
<td><strong>Major differences in 500</strong></td>
<td>PHB/Lysine</td>
<td>butanol/isobutanol</td>
</tr>
<tr>
<td><strong>most frequent words</strong></td>
<td>PTS</td>
<td>HPLC</td>
</tr>
<tr>
<td></td>
<td><em>B. subtilis</em></td>
<td>cyanobacteria</td>
</tr>
<tr>
<td></td>
<td>hydrogen</td>
<td>Lipid</td>
</tr>
<tr>
<td></td>
<td><em>P. Pastoris</em></td>
<td><em>C. glutamicum</em></td>
</tr>
</tbody>
</table>

Table 7.2. Development trend of ‘metabolic engineering’ during 2000 ~ 2009 and 2010 ~ 2015, similarity 81%
<table>
<thead>
<tr>
<th>Terms</th>
<th>Biofuel</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2000 ~</td>
</tr>
<tr>
<td><strong>Most frequent words</strong></td>
<td>Production</td>
</tr>
<tr>
<td><strong>Major differences in 500</strong></td>
<td>Lignocellulosic hemicellulose Emissions policy Economic/market Feedstock</td>
</tr>
<tr>
<td><strong>most frequent words</strong></td>
<td></td>
</tr>
</tbody>
</table>

Table 7.3. Development trend of ‘biofuel’ during 2000 ~ 2009 and 2010 ~ 2015 similarity 72.4%
Figure 7.1a. Total number of journals in the database at different time

Figure 7.1b. Total number of papers in the database increase at different time
Figure 7.2. Structure of nxml file
Figure 7.3. Word cloud of ‘metabolic engineering’
Figure 7.4. Word cloud of ‘environmental engineering’
Figure 7.5. Word cloud of ‘synthetic biology’
Figure 7.6. Word cloud of ‘systems biology’
Figure 7.7. Word cloud of ‘metabolic flux’
CHAPTER EIGHT

CONCLUSIONS AND FUTURE PERSPECTIVES

8.1 Conclusions

In this dissertation, I first employed fluxomics tools (FBA and $^{13}$C-MFA) to investigate the metabolisms of isobutanol-producing E. coli strains in chapter two. $^{13}$C-MFA results indicated that isobutanol production reshaped the central metabolism with increased activities in Pentose Phosphate pathway (supply more NADPH for isobutanol synthesis), and glyoxylate shunt (reserve more carbon). To gain an overview on how the factor of oxygen concentration, P/O ratio, and maintenance energy affects isobutanol production, we set up an integrated flux model (use $^{13}$C-MFA flux values to constrain genome scale FBA) to test the sensitivity of each factor. The simulation result revealed that the maintenance energy played a more important role in affecting isobutanol yield than P/O ratio and oxygen flux. Further, we found a ‘cliff’ in isobutanol yield landscape which can be triggered by increased maintenance energy, decreased P/O ratio or decreased oxygen supply. Discovery of this cliff explained many failures observed in isobutanol experiments and scale-up processes. Besides, we also quantify the impacts of yeast extract and the results show that supply of rich nutrients such as yeast extract can efficiently relieve the intracellular crisis in carbon and energy resource, thereby can significantly boost isobutanol production without directly contributing to the product. The limitation of this work is industrial isobutanol production mainly focus on the non-growth phase, which central metabolites tracing, fast quenching, as well as dynamics $^{13}$C-MFA are required to resolve cellular metabolism at this phase.
We expanded the analysis to other biofuel producing cases in chapter three. FBA simulations indicated that alcohol (i.e., ethanol, isobutanol) production is more robust to P/O ratio change (efficiency of energy metabolism) than fatty acid production in \textit{E. coli} strains. In addition, we took an overview of those successful and failure cases in biotechnology industry and realized that avoiding extensive genetic modification is important for high yield and strain stability. Heterogeneous plasmid or enzyme expression introduces extra metabolic burden, leading to shifted central metabolism, and even imbalance between carbon and energy metabolism (Yin-Yang Balance). We proposed several approaches to solve this problem, such as employment of fluxomics tools (i.e., \textsuperscript{13}C-MFA) to decipher energy metabolism, relying on native pathways, performing minimal engineering, \textit{etc}. Based on the Yin-Yang theory, we tried to insert \textit{Vitreoscilla} hemoglobin (\textit{vhb}) gene into fatty acid producing strains to improve its oxygen uptake ability and thus fatty acid production, which is discussed in chapter four.

Chapter four

Besides energy metabolism related projects, I also developed a series of computational tools for fluxomics studies and literature analysis. In chapter five, I rebuilt the MicrobesFlux platform, moved it from a share server at Washington University to a commercial server (Amazon EC2 server). In addition, I enhanced the functionality of MicrobesFlux by including much more species (previously: 1304 species; currently: 3192 species), supporting the SBML file format, having unlimited storage space. Further, I participated in the development of MATLAB based open-source \textsuperscript{13}C-MFA software (WUFlux). In addition, I designed and built several website platforms (fluxomics.net, 13cmfa.org) for fluxomics studies, also we shared all fluxomics tools used in our group on the website for free.
In chapter six, we built a computational platform that can calculate microbial metabolism very quickly based on user input. This computational platform named MFlux (MFlux) is developed based on an integration of machine learning, constraint programming, and quadratic programming. Through grid searching, we chose Support Vector Machine as machine learning algorithm and optimize its parameters. Quadratic programming can adjust predicted flux profile to satisfy the stoichiometry constraints and constraint programming is used to avoid non-sense inputs. To sum up, we build up the correlations between genetic/environmental factors and central metabolic fluxes.

In chapter seven, I developed a Big Data based framework which can perform fast literature analysis based on user inputs. We first employed the text mining approach to extract information from full text papers. Subsequently, we upload all information into Google Cloud server and set up a fast search tool through Google BigQuery. Finally, through BigQuery search plus searching results processing, we get meaningful information from literature database. We performed several case studies, and the results turned to be fast (whole process less than 2 min), reliable (results repeatable), and informative. The limitation of this work is that current literature database only contains 1.1 million papers focusing on biomedical research; and we need more information to avoid bias in analysis.

8.2. Future directions to solve intracellular energy crisis

To circumvent the energy bottlenecks within microbial cell factories, we propose the following approaches in addition to those already mentioned in Chapter three:

1. Improve energy efficiency by engineering respiration metabolism
In aerobic metabolism, energy generation mainly comes from ATP synthesis through oxidative phosphorylation (respiration). However, in real cases, respiration rates in many strains are far below theoretical maximum (Varma and Palsson 1994; Wu et al. 2015; Sauer and Bailey 1999). Thereby, improving oxidative respiration efficiency is an efficient manner to enhance energy supply. The first successful case of knocking out inefficient respiration metabolism was reported by Zamboni and Sauer, to enhance riboflavin production in *Bacillus subtilis* (Zamboni et al. 2003). In this work, knockout of cytochrome *bd* oxidase also leads to a 40% reduction of cellular maintenance energy. Afterwards, this strategy has been adopted and combined with other strategies, to enhance the yield of other products (e.g., N-acetylglucosamine) in *B. subtilis* (Liu et al. 2014). Observations of reduced maintenance energy were also reported in model species *E. coli* (*ndh* knockout) (Calhoun et al. 1993) and *C. glutamicum* (*cydAB* knockout) after eliminating the energy metabolism component of low efficiency (Kabus et al. 2007). Further, the cytochrome *bd* knockout mutant was reported to enhance lysine production by ~12% in *Corynebacterium glutamicum* (Kabus et al. 2007). This strategy may apply to species with several sets of respiration chains with different efficiency. Notably, successful applications of this strategy are also closely related with other factors, such as oxygen concentration, medium composition and etc. (Kabashima et al. 2009).

2. **Utilization of other energy sources**

As a traditional energy source, hydrogen can be utilized by a broad range of microbes (B Friedrich and Schwartz 1993; Petersen et al. 2011), Utilization of hydrogen as the energy supplier in industrial fermentation is not preferred, albeit the well-known syngas fermentation. This is due to those undesired properties of hydrogen gas such as low solubility, easy-to-leak,
and explosive. Compared with hydrogen, formate is a better source of energy supply in terms of the uptake efficiency.

The utilization of formate by microbes as extra energy source was discovered more than thirty years ago. In 1983, Bruinenberg et al. discovered that Candida utilis can uptake formate as an additional energy source in the presence of glucose (Bruinenberg et al. 1983). Utilization of formate leads to an increased biomass yield. Meanwhile, similar phenomenons were also observed in Hansemula polymorpha by Babel et al. and in Pichia pastoris by Hazeu et al. (Babel et al. 1983; Hazeu and Donker 1983). Saccharomyces cerevisiae CBS 8066 also joined this list of formate utilization species. However, formate utilization did not make any improvement in biomass yield of S. cerevisiae (Bruinenberg et al. 1985). It was then realized that utilization of formate normally requires the functionality NAD-dependent formate dehydrogenase (FDH, EC 1.2.1.1). The strategy of employing formate as extra energy source has been extended to other species with FDH, such as oleaginous yeasts (Cryptococcus curvatus, Rhodotorula glutinis, and Lipomyces starkeyi) for improving lipid production (Lian et al. 2012), Penicillium chrysogenum for enhancing penicillin G productivity (Harris et al. 2007), and Bacillus thuringiensis for promoting thuringiensin yield (Zhi et al. 2007). Formate can also be generated through an electrochemical process, to feed engineered Ralstonia eutropha H16 strain to produce isobutanol and 3-Methyl-1-butanol (Li et al. 2012). On the other side, heterologous expression of an fdh gene enables formate usage in those species without this gene. For instance, after insertion of the fdh gene into the chromosome, Corynebacterium glutamicum was able to utilize formate and produce 20% more succinate anaerobically in the presence of glucose. Formate was used as the NADH and CO2 donor in this case (Litsanov et al. 2012). In yet another case, fdh was introduced into a succinate-producing E. coli strain, leading to significantly reduced byproduct
formate and improve succinate yield and productivity (Balzer et al. 2013). Metabolic burden of *fdh* overexpression may offset its benefit; thus, careful consideration is necessary at the stage of strain design.

\[
\text{Formate + NAD} \rightarrow \text{CO}_2 + \text{NADH}
\]

Photosynthetic microorganisms (cyanobacteria and microalgae) convert CO\(_2\) to useful products with light as the energy source. The list of their products has been greatly boosted with the advent of advanced genetic tools (Wijffels et al. 2013). However, scale-up photosynthetic microbial process to industrial production has been severely hindered by a limited range of light penetration, which leads to a series of problems such as high-cost photobioreactor, low cell density, and cost-inefficient harvesting. To circumvent these problems, a photomixotrophic strategy has been proposed (You et al. 2015). Under light and glucose sufficient condition, *Synechocystis* sp. PCC 6803 is able to consume both CO\(_2\) and glucose for biomass production which potentially leads to higher biomass density.

### 3. Minimize maintenance energy in host strain

During industrial fermentation, cellular maintenance energy is released out as heat, leading to increased temperature of fermentation broth, which is undesired for the whole process. Further, low maintenance energy requirement indicates more energy allocated on biomass & product synthesis (Sauer et al. 1996). Thereby, hosts of low maintenance energy are preferred. In a comparative study, cellular metabolisms of several bacilli strains (both wild type and mutant) close to *Bacillus subtilis* were investigated. And the result show that the *B. licheniformis* T380B strain has the lowest maintenance energy (0.20 mmol/g*h) of all strains analyzed in that work. Therefore, the authors consider it as a potential host to replace *B. subtilis* (maintenance energy 0.39 mmol/g*h) for vitamin and other chemical production (Tannler et al. 2008).
On the other side, cellular maintenance energy is positively correlated with incubation temperature (Price and Sowers 2004; Lever et al. 2015). From energy point of view, lower temperature is preferred for fermentation process, in the sake of lower maintenance energy and enhanced strain stability. However, temperature is an essential factor affecting many processes within cellular metabolism (e.g., enzymes kinetics, regulation). In practice, a trade-off between production rate and strain stability requires careful consideration during temperature control (Dunn-Coleman et al. 1992).

8.3 Personal views on the future of microbial cell factories

8.3.1. Limitation of microbial systems

Each closed system has its limitation. For instance, enzymatic systems are facing the trade-off between its kinetic efficiency ($K_{cat}/K_{m}$) and thermostability: it is very challenging to obtain enzymes with both good thermostability and high catalytic efficiency (Ye et al. 2012; Romero and Arnold 2009). In a similar manner, microbial systems have a series of physical limitations: the trade-off between cell membrane surface area/volume and material exchange efficiency, the limited cell membrane surface area decides the upper limit of nutrient and oxygen uptake rate (Zhuang et al. 2011), pose a further restriction on the total carbon/energy available within a cell. Taken all limiting factors together before the design, we are able to realize what is impossible even before many failures after extensive engineering.

8.3.2. The success of distributed system in computational system shed lights on microbial systems

Computational systems have their limitation: the CPU frequency is strictly limited by the speed of signal travel (light of speed), therefore, after we had 3GHz CPU more than ten years (2002), it
is difficult to get more improvements in CPU clock. Expensive multi-core high performance working stations were the first choice to handle those extensive computation, however, are being replaced by distributed computational systems (GDFS, Google Distributed File System or HDFS, Hadoop Distributed File System, both are based on MapReduce algorithm) due to high costs and low robustness to error. Reliability (robustness) is an important factor deciding the commercial values of any systems, and that is the reason distributed computational systems wins the battle with working stations. Similarly, extensive genetic modification on microbial cells bring extra burdens on cell metabolism which leads to increased instability of cell itself, as well as its poor reliability, which has been verified by many failed bioprocess scale-up. In a similar manner, microbial distributed systems (e.g., coculture (Zhou et al. 2015), integrated bio-chem process (Xiong et al. 2014)) are still in its infant phase. The key point for successful microbial distributed systems is to have a universal framework (similar as MapReduce) to improve the reliability of the system substantially, which current synthetic biology or metabolic engineering does not solves.

8.4. Reference


Petersen, J. M., Zielinski, F. U., Pape, T., Seifert, R., Moraru, C., Amann, R., Hourdez, S., Girguis, P. R., Wankel, S. D., Barbe, V., Pelletier, E., Fink, D., Borowski, C., Bach, W. and


Appendix I

1. List of courses taken and to be taken with grades

Courses finished:
EECE501 Transport Phenomena in Energy, Environmental, and Chemical Engineering
EECE5404 Combustion Phenomena
EECE548 Environmental Organic Chemistry
EECE503 Kinetics and Reaction Engineering Principles
EECE596 Metabolic Engineering
EECE517 Partial Differential Equations
BIOL5014 Biotech Industry Innovators
EECE590 Energy and Environmental Economic Decision-Making

Courses to be transferred from master program (Virginia Tech):
CHE5984 Engineering Mathematics
STAT5605 Biometry 1
MATH5515 Mathematical Methods for Modeling and Simulation of Biological Systems

2. TA experiences

CHE473A Chemical Engineering Laboratory (12 fall)
EECE262 Introduction to Environmental Engineering (13 spring)
CHE478A Process & Product Design (14 spring)

Workshop certification:
Grading and responding to students’ concerns about grades
Improving presentation skills

3. Academic achievements

Conference experience:
2014 Annual Meeting of American Institute of Chemical Engineers (AIChE) Poster
2012 Annual Conference of Sustainable Nanotechnology Organization (SNO) Poster
2012 Annual Meeting of American Institute of Chemical Engineers (AIChE) Poster
Publication:

L. He, S. G. Wu, and Y. J. Tang, WUFLux: an open-source MATLAB based software for $^{13}$C-MFA, under review by *BMC Bioinformatics*

S. G. Wu, A. Varman, L. He, and Y. J. Tang, Evaluating physiological state of engineered e. coli strains by isotopomer constrained flux balance analysis, *in preparation*


Awards:

National Institute for Mathematical and Biological Synthesis (NIMBioS) Visiting Graduate Fellowship, 2013
Student Award, 1st Annual Conference of Sustainable Nanotechnology Organization (SNO), Arlington, VA, 2012

Appendix II Supporting information of Chapter 6

S1. Source code

```
clp.py

def process_species_db(File):
    """Load the species database which is Suppiliment Information I

    Format
    ========
    Fields separated by tab
    Species Species name Oxygen condition   Substrate uptake rate
    upper bound (mmol/gDW*mol ) 1  2  3  4  5  6  7  8  9  10
    11  12  13  14 Growth rate upper bound (h-1) Reference
    1   Escherichia coli  1,3,2  20  Y  Y  Y  Y  Y  Y  Y  Y
    Y  Y  Y  Y  Y  N  1.2  1
    2   Corynebacterium glutamicum  1,3,2  40  Y  Y  Y  Y  Y  Y
    Y  Y  N  Y  N  Y  N  1  2

    Returns
    ========
    DB: A list of tuples. Each tuple is (species, Oxygen, rate,
    Carbon1, Carbon 2, ..., Carbon 14, Growth_rate_upper)
    Oxygen itself is a string, e.g., "1,2,3"
    """

    Carbon_sub = {"Y":True, "N":False}
    DB = []
    with open(File, "r") as F:
        F.readline() # skip the header
        for Line in F:
            Field = Line.split("\t")
            #
            [Species, Substrate_rate] = map(int, [Field[0], Field[3]])
            Oxygen = Field[2] #map(int, Field[2].split(",")
            Carbon_src = [ Carbon_sub.get(x, False) for x in
            Field[4:4+13+1] ]
            Growth_rate_upper = Field[4+14]
            DB.append(tuple([Species, Oxygen, Substrate_rate]+
            Carbon_src + [Growth_rate_upper]))

    return DB

```
def species_db_to_constraints(DB, Debug=False):
    """Turn the species DB into a CSP problem (constraints only, no variable ranges)"

    Parameters
    =========
    DB: list of tuples
    Each tuple is (species, Oxygen, rate, Carbon1, Carbon 2, ..., Carbon 14)
    Oxygen itself is a tuple, e.g., (1,3)

    Returns
    =======
    problem: an instance of python-constraint
    containing only constraints but no variable domains

    Notes
    ======
    the problem has a solution if any of the rules set in species database is VIOLATED.
    In other words, if the problem has solution, then the input does NOT make sense.

    ""
    import constraint # python-constraint module
    problem = constraint.Problem() # initialize the CSP problem

    # create variables
    # problem.addVariable("Species", range(1,41+1))
    # problem.addVariable("Substrate_rate", range(0, 100+1))
    # problem.addVariable("Oxygen", [1,2,3])
    # for i in xrange(1, 14+1):
    #     problem.addVariable("Carbon"+str(i), [True, False]) # create one variable for each carbon source
    # This part should be from user input

    # add constraints, where each entry in DB is a constraint.
    # create the lambda functions
    All_vars= ["Species", "Substrate_rate", "Oxygen"] + 
    ["Carbon"+str(i) for i in xrange(1, 14+1)] + ["Growth_rate_upper"]
    for Entry in DB:
        Oxygen_values = Entry[1] # as string
        Foo = "lambda "
        Foo += ", ".join(All_vars) # done with listing all variables
        Foo += " : "
        Logic_exp = ["Substrate_rate<=" + str(Entry[2]), "Species==" + str(Entry[0]), "Growth_rate_upper<=" + str(Entry[4+13])]

        for i in xrange(3, 3+14): # carbon sources
            if not Entry[i]: # only use false ones to create the constraint
                Logic_exp += ["Carbon"+str(i) + str(Entry[i])]

        Foo += "(" + "\n        Foo += "
        Foo += "\n        Foo += "\n        Foo += "\n        Foo += "\n        Foo += "\n        Foo += "\n        Foo += "\n        Foo += "\n        Foo += "\n        Foo += "\n        Foo += "\n        Foo += "\n        Foo += "\n        Foo += "\n        Foo += "\n        Foo += "\n        Foo += "\n        Foo += "\n        Foo += "\n        Foo += "")

        problem.addConstraint(Foo

Logic_exp.append( ( All_vars[i] + "==" + str(Entry[i]) ) )

Logic_exp.append( ( "Oxygen in [" + Oxygen_values + "]" ) )

Logic_exp = " and ".join(Logic_exp)
Logic_exp = "not (" + Logic_exp + ")"  # De Morgan's Law
if Debug:
    print Logic_exp
problem.addConstraint(eval(Foo + Logic_exp), tuple(All_vars))

return problem  # just return one solution, if no solution, return
is NoneType

def input_ok(problem, Vector):
    """Turn user inputs into domains of variables for the CSP problem
    and then solve.
    Parameters
    =========
    problem: a python-constraint instance with constraints built
    Vector: the feature vector, float numbers, [Species, Reactor,
    Nutrient, Oxygen, Method, MFA, Energy, Growth_rate,
    Substrate_uptake_rate] + ratio of 14 carbon sources in the order:
    "glucose", "fructose", "galactose", "gluconate", "glutamate",
    "citrate", "xylose", "succinate", "malate", "lactate", "pyruvate",
    "glycerol", "acetate", "NaHCO3"
    Notes
    ======
    In current formulation, the problem has a solution if any of the
    rules set in species database is VIOLATED.
    In other words, if the problem has solution, then the input does
    NOT make sense.
    Example
    =======
    >>> import clp
    >>> DB = clp.process_species_db("SI_1_species_db.csv")
    >>> P = clp.species_db_to_constraints(DB)
    >>> Vector = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.72, 10.47, 1.0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.0]
    >>> print clp.input_ok(P, Vector)
    True
    >>> P.reset()  # another test, violating the carbon source it takes
    >>> P = clp.species_db_to_constraints(DB)
    >>> Vector = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.72, 17, 1.0, 0,
    0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0.0]
    >>> print clp.input_ok(P, Vector)
    False
    >>> P.reset()  # another test, violating growth rate upper boundary
    >>> P = clp.species_db_to_constraints(DB)
```python
>>> Vector = [1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 1.0, 0.72, 10.47, 1.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0]
>>> print clp.input_ok(P, Vector)
False

problem.addVariable("Species", [Vector[0]])
problem.addVariable("Substrate_rate", [Vector[8]])
problem.addVariable("Oxygen", [Vector[3]])
problem.addVariable("Growth_rate_upper", [Vector[7]])
for i in xrange(1, 14+1):
    problem.addVariable("Carbon"+str(i), [True if Vector[i+8]>0 else False])  # create one variable for each carbon source

Solutions = problem.getSolution()

if Solutions == None:# no a single solution, pass test
    return True
else:
    return False
```

```python
compute.py
#!/usr/bin/env python

import cgi
form = cgi.FieldStorage() # instantiate only once!

import cgitb; cgitb.enable()

# Feature_names =  
# Feature_names =  
# Feature_names =  
# Feature_names =  
# Feature_names =  
Feature_names =  

Features = {
"Energy":1.0, "MFA":1.0, "Substrate_other":0.0}

# Avoid script injection escaping the user input
#Purpose = "sdfsd"

print "Content-Type: text/html"
print
```
print """
<html>
<head><title>Result of Influx analysis </title></head>

<body>
<h2>Parameters entered:</h2>

# Process the form values
for Feature_name in Feature_names:
    Feature_value = form.getfirst(Feature_name)
    Feature_value = cgi.escape(Feature_value)
    Features[Feature_name] = float(Feature_value) # convert all string
to numbers

    print """
    %s is %s,
    """ % (Feature_name, Feature_value)

import libflux
Vector, Substrates = libflux.process_input(Features)
Boundary_dict = libflux.process_boundaries(form, Substrates)

#libflux.test("hello, world")
Influxes = libflux.predict(Vector, Substrates, Boundary_dict) # use
the feature vector to predict influx values

print """
<p><a href="index.html">Go back to submission page</a></p>

<hr>
<p>This project is supported by National Science Foundation. <a href="http://www.nsf.gov/awardsearch/showAward?AWD_ID=1356669">More
info</a> <br>
Information on this website only relects the perspectives of the
individuals.<br>
</p>
</body>
"""

---------------------------------------------

get_model.py

# extract the training data from spreadsheet

from collections import defaultdict
import cPickle
import numpy
from sklearn import cross_validation, preprocessing, grid_search
from sklearn.neighbors import KNeighborsRegressor
from sklearn.svm import SVR
from sklearn.tree import DecisionTreeRegressor

class RegressionModel(object):
    """A help class to store model's name and the actual model."""
    def __init__(self, name, **kwargs):
        self.name = name
        self.model = eval(name)(**kwargs)
    def __str__(self):
        return self.name

class RegressionModelFactory(object):
    """A factory to create new instances of RegressionModel."""
    def __init__(self, name, **kwargs):
        self.name = name
        self.kwargs = kwargs
    def __str__(self):
        return "{} ({})".format(self.name, self.kwargs)
    def __call__(self):
        """Create a new RegressionModel instance each time."""
        return RegressionModel(self.name, **self.kwargs)

def shuffle_data(Training_data):
    """Shuffle the order of data in training data

    Shuffle by scrambling the index

    (New)_training_data: a dict, keys are EMPs (e.g., v1, v2, etc.),
    values are 2-tuples (Feature, Label), where
    Feature is a 2-D list, each sublist is 24-D feature
    vector for one sample
    and
    Label is a 1-D list, labels for all samples.
    """
    import random
    New_training_data = {}
    for i, (Feature_vector, Labels) in Training_data.iteritems():
        Num_Samples = len(Feature_vector)

if Num_Samples != len(Labels):
    print "Error! Inconsistent numbers of Features Vectors and Labels"
Shuffled_index = range(Num_Samples)
random.shuffle(Shuffled_index)
    New_Feature_vector = [Feature_vector[j] for j in Shuffled_index]
    New_Label = [Labels[j] for j in Shuffled_index]

    New_training_data[i] = ([New_Feature_vector, New_Label])

return New_training_data

def read_spreadsheet(filename):
    """Turn spreadsheet into matrixes for training

    Returns
    ======
    training_data: a dict, keys are EMPs (e.g., v1, v2, etc.),
    values are 2-tuples (Feature, Label), where
    Feature is a 2-D list, each sublist is 24-D feature vector for one sample
    and
    Label is a 1-D list, labels for all samples.

    Notes
    ======
    EMPs are N.A. for some samples, training features were dropped for them.
    That's why we need one training feature matrix for each EMP.

    We have 29 influxes values to predict and thus the index/key for training_data goes from 1 to 29

    AA is 26 for 0-index
    BA is 32 for 0-index
    ""

    training_data = {}
    for i in range(1, 29+1):# prepare the data structure
        training_data[i] = ([],[]) # the 1st list is the features and the 2nd the labels for i-th influx
    
    reports = defaultdict(list)
    with open(filename, 'r') as f:
        for i, line in enumerate(f.readlines(), 1):
            line = line.strip()
            line = line.split("\t")
            vector = line[2:26+1] # training vector, from Species (C) to Other carbon (AA).
key = ",\ ,\ .join(vector)
reports[key].append(i)

if "" in vector:
    vector.remove(""
if not vector :
    print line
    exit()

labels = line[26+3: 26+3+26+5] # AD to BF, v1 to v29
    #
    #            print Labels
try:
    vector = map(float, vector)
except ValueError:
    print vector

# Now create the dictionaries we need, one dictionary for each influx
for i in range(1, 29+1):
    label = labels[i-1]
    try:
        label = float(label)
    except ValueError:
        #                    print Label, "=>"
        #                    print Line
        continue # this label for this influx is not numeric
    training_data[i][0].append(vector) # add a row to feature vectors
    training_data[i][1].append(float(label)) # add one label

print("checking duplicate lines...")
for k, v in reports.iteritems():
    if len(v) > 1:
        print("line number: {}".format(v))
print("Done."
return training_data

def one_hot_encode_features(training_data):
    """Use one-hot encoder to represent categorical features
    Feature from 1 to 7 are categorical features:
    Species, reactor, nutrient, oxygen, engineering method, MFA and extra energy
    """
    import numpy
encoded_training_data, encoders = {}, {}
for vid, (vectors, targets) in training_data.items():
    encoder = preprocessing.OneHotEncoder()
    vectors = numpy.array(vectors) # 2-D array
    encoded_categorical_features =
        encoder.fit_transform(vectors[:, 0:6+1])
    encoded_categorical_features =
        encoded_categorical_features.toarray()
    encoded_vectors =
        numpy.hstack((
            encoded_categorical_features, vectors[:, 6+1:]))
    encoded_training_data[vid] = (encoded_vectors, targets)
    encoders[vid] = encoder
return encoded_training_data, encoders

def standardize_features(training_data):
    """Standarize feature vectors for each influx

    Later, a new feature vector X for i-th influx can be normalized as:
    Scalers[i].transform(X)
    ""

    std_training_data, scalers = {}, {}
    for vid, (vectors, labels) in training_data.items():
        vectors_scaled = preprocessing.scale(vectors)
        std_training_data[vid] = (vectors_scaled, labels)
        scalers[vid] = preprocessing.StandardScaler().fit(vectors)
    return std_training_data, scalers

def train_model(training_data, Parameters):
    """Train a regression model for each of the 29 influxes

    Returns
    =========
    Models: dict, keys are influx indexes and values are regression models
    parameters: dict, keys are intergers 1 to 29, values are dicts, such as
    'epsilon': 0.01, 'c': 100.0, 'gamma': 0.001, 'kernel': 'rbf'

    Notes
    ======
    Parameters are not in use. Now use same parameters for all v's.
    ""
    models = {}
    for i in range(1, 29+1):
        vectors, label = training_data[i]
Parameter = Parameters[i]
model_gen = RegressionModelFactory("SVR", **Parameter)
#    model_gen = RegressionModelFactory("SVR", kernel="linear", C=0.1, epsilon=0.01)
#    model_gen = RegressionModelFactory("KNeighborsRegressor", n_neighbors=10, weights="distance")
model = model_gen().model
model.fit(vectors, label) # train the model
models[i] = model
return models

def cross_validation_model(training_data, model_gen, Folds, N_jobs):
    """Do a cross validation on a model using the given training data.

    :param training_data: A dict with keys as v, and values as [vectors, label].
    :param model_gen: A RegressionModel generator or a list of that.
    :param Folds: number of CV folds
    """

    import sklearn
    #    print("model: {}\).format(model_gen))
    print("\tscore\naccuracy")
folds = Folds
for i in range(1, 29 + 1):
    vectors, label = training_data[i]
    label = numpy.asarray(label)
    if type(model_gen) == dict:
        model = model_gen[i]()
    else:
        model = model_gen()
    # allow shuffleSplit on dataset
    print len(label)
    if Folds < 1:
        folds = sklearn.cross_validation.ShuffleSplit(len(label))
scores = cross_validation.cross_val_score(model.model, vectors,
                                      cv=folds,
                                      scoring="mean_squared_error",
                                      n_jobs = N_jobs
                                      scoring="r2"
                                      )
    print("\{}\t{} (+/- {})
          .format(i, scores.mean(), scores.std() * 2))
    # This needs to scaled back to real range of fluxes.

def grid_search_cv(training_data, model_gen, params, SCORINGS,
CORE_NUM, FOLDS):
Do a grid search to find best params for the given model.

:param training_data: A dict with keys as v, and values as [vectors, label].
:param model_gen: A RegressionModel generator.
:param params: All parameters the grid search needs to find. It's a subset of all the optional params on each model. i.e. for KNeighborsRegressor model, it's a subset of

```
{
  "n_neighbors": [1, 5, 10, ...],
  "weights": ["uniform", ...],
  "algorithm": ["auto", ...],
  "leaf_size": [30, 50, ...],
  "p": [2, 5, ...],
  "metric": ["minkowski", ...],
  "metric_params": [ ...],
}
```

print("model: {}").format(model_gen)
print("v
tscoring	best_score	best_params")
for i in range(1, 29 + 1):
    vectors, label = training_data[i]
    model = model_gen()
    for scoring in SCORINGS:
        clf = grid_search.GridSearchCV(model.model, params,
            scoring=scoring, n_jobs=CORE_NUM, cv=FOLDS)
        clf.fit(vectors, label)
        print("{}\ts{}	{}	{}".format(i, scoring, clf.best_score_,
            clf.best_params_))

def grid_search_tasks(std_training_data):
    """One function to run grid search on different regressors"

    CORE_NUM: int, number of CPU cores to be used
    FOLDS: int, number of folds for cross validate

    """
    import numpy
    knn_model_gen = RegressionModelFactory("KNeighborsRegressor",
        n_neighbors=10, weights="distance")
    svr_model_gen = RegressionModelFactory("SVR", kernel="linear",
        C=10, epsilon=0.2)
    dtree_model_gen = RegressionModelFactory("DecisionTreeRegressor",
        random_state=0)

    KNN_PARAMS = {

216
"n_neighbors": range(1, 16),
"weights": ["distance", "uniform"],
"algorithm": ["ball_tree", "kd_tree", "brute"],
"metric": ["euclidean", "chebyshev", "minkowski"],

SVR_PARAMS = {
    "C": 10.0 ** numpy.arange(-4, 4),
    "epsilon": [0., 0.0001, 0.001, 0.01, 0.1],  # experience: epsilon>=0.1 is not good.
    "kernel": ["linear", "rbf", "poly", "sigmoid"],
    "gamma": 10.0 ** numpy.arange(-4, 4),
}

DTREE_PARAMS = {
    "criterion": ["mse"],
    "splitter": ["best", "random"],
    "min_samples_split": range(2, 16),
    "min_samples_leaf": range(1, 16),
    "max_features": ["sqrt", "log2"],
    "random_state": [0, 1, 10, 100],
}

SCORINGS = ["mean_squared_error", "mean_absolute_error"
]

TRAINING_PARMAS = [
    (knn_model_gen, KNN_PARAMS),
    (svr_model_gen, SVR_PARAMS),
    (dtree_model_gen, DTREE_PARAMS),
]

FOLDS = 10
CORE_NUM = 32

[grid_search_cv(std_training_data, k, v, SCORINGS, CORE_NUM, FOLDS) for k, v in TRAINING_PARMAS]

def cv_tasks(std_training_data, Folds, N_jobs, Label_scalers, Parameters):
    """Cross-validation on all v's
:param Folds: number of CV folds
:param N_jobs: number of CPU cores
:param label_scaler: dict, keys are fluxes and values are sklearn scaler objects
:param Parameters: dict, keys are fluxes and values are parameters for all fluxes

```python
import sklearn
knn_model_gen = RegressionModelFactory("KNeighborsRegressor",
n_neighbors=10, weights="distance")
#     dtree_model_gen = RegressionModelFactory("DecisionTreeRegressor",
#     random_state=0)

if Parameters != None: # need to create one instances for one flux
    svr_model_gen = {}
    for i in xrange(1, 29+1):
        svr_model_gen[i] = RegressionModelFactory("SVR",
**{Parameters[i]})
else: # same set of parameters for all SVR models.
    svr_model_gen = RegressionModelFactory("SVR", kernel="linear",
C=0.1, epsilon=0.01)

Classifier_models = [
#     knn_model_gen,
    svr_model_gen,
#     dtree_model_gen,
]

[cross_validation_model(std_training_data, m, Folds, N_jobs) for m
in Classifier_models]

def svr_training_test(std_training_data, Parameters,
Label_scalers=None):
    
    Parameters
    =========
    std_training_data: dict, keys are vID, values are tuples (vector, label)
                   each vector is 2-D array and label is a 1-D array

    Parameters: dict, keys are integers 1 to 29, values are dicts,
             such as

              'epsilon': 0.01, 'c': 100.0, 'gamma': 0.001, 'kernel':
              'rbf'

    Label_scalers: dict, keys are int 1 to 29, value sare sklearn
                    scaler objects
from numpy import square, mean, sqrt
Models = train_model(std_training_data, Parameters)
Influxes = {}

for vID, Model in Models.iteritems():
    (Vectors_for_this_v, Label_for_this_v) = std_training_data[vID]
    Label_predict = Model.predict(Vectors_for_this_v)
    if Label_scalers != None:
        Label_predict = Label_scalers[vID].inverse_transform(Label_predict)
        Label_for_this_v = Label_scalers[vID].inverse_transform(Label_for_this_v)
    MSE = Label_predict - Label_for_this_v
    # if vID==2:
    #     print Label_predict
    #     print Label_for_this_v
    #     print MSE
    MSE = sqrt(mean(square(MSE)))

    print "\t\t".join(map(str, [vID, MSE , max(Label_for_this_v), min(Label_for_this_v) ])) + "\\n"
    # for i, j in enumerate(list(MSE)):
    #     print i+1, j
    # print list(square(MSE))
    # print Label_predict
    # break

def _validate_training_data(training_data):
    reports = []
    for _, d in training_data.iteritems():
        report = defaultdict(list)
        vectors = d[0]
        for i, v in enumerate(vectors):
            key = ", ".join(map(str, v))
            report[key].append(i)
        # only keep duplicated rows
        report_ = {k: v for k, v in report.iteritems() if len(v) > 1}
        reports.append(report_)

    return reports

def label_std(Training_data, Method="Norm"):
    """standardize the labels in training data
    training_data: a dict, keys are EMPs (e.g., v1, v2, etc.),
    values are 2-tuples (Feature, Label), where
    Feature is a 2-D list, each sublist is 24-D feature
    vector for one sample
    and
    Label is a 1-D list, labels for all samples.
    """
Label_scalers: dict, keys are vIDs and values are sklearn.preprocessing.MinMaxScaler instances for 29 influxes

sklearn's preprocessing MixMaxScaler does column-wise Minmax scaling.
Since influxes have different number of instances, we must loop thru the 29.

```python
import sklearn
Label_scaled_data = {}
Label_scalers = {}
if Method == "None": # No label std needed
    return Training_data, None

for vID, (Vector, Label) in Training_data.iteritems():
    if Method == "Norm":
        Label_scaler =
        sklearn.preprocessing.StandardScaler().fit(Label)
        # Label_scaled = sklearn.preprocessing.scale(Label)  # option 1 of standardization
    elif Method == "MinMax":
        Label_scaler =
        sklearn.preprocessing.MinMaxScaler().fit(Label)
    else:
        print "Unrecognized label standardization method "
        Label_scaled = Label_scaler.transform(Label) # Option 2, MinMax scaler

    Label_scaled_data[vID] = (Vector, Label_scaled)
    Label_scalers[vID] = Label_scaler

return Label_scaled_data, Label_scalers
```

def load_parameters(File):
    """Load a parameter file from grid search print out

    The format of grid search print out:
    checking duplicate lines
    model: SVR ({'epsilon': 0.2, 'C': 10, 'kernel': 'linear'})
    v scoring  best_score best_params
    1 mean_squared_error -0.00462588529703 {'epsilon': 0.01, 'C': 100.0, 'gamma': 0.001, 'kernel': 'rbf'}
    2 mean_squared_error -0.0103708930608{'epsilon': 0.01, 'C': 1000.0, 'gamma': 0.0001, 'kernel': 'rbf'}
    3 mean_squared_error -0.00713773093885 {'epsilon': 0.01, 'C': 1000.0, 'gamma': 0.0001, 'kernel': 'rbf'}
    4 mean_squared_error -0.0115793576617{'epsilon': 0.001, 'C': 1000.0, 'gamma': 0.0001, 'kernel': 'rbf'}
    """
import re

Parameters = {}

with open(File, 'r') as F:
    F.readline() # Skip first line
    F.readline() # Skip second line
    F.readline() # Skip 3rd line
    for Line in F.readlines():
        [v, _, _, Parameter] = Line.split("\t")
        v = int(v)
        exec "Parameter = " + Parameter
        Parameters[v] = Parameter

return Parameters

def test_label_std():
    """Test the accuracy on labels using different label std methods

    The 3 methods are: no std, normalization, MinMax. We will study RMSE under different normalization
    ""
    training_data = read_spreadsheet("wild_and_mutant.csv")
    training_data = shuffle_data(training_data)
    encoded_training_data, encoders =
    one_hot_encode_features(training_data)
    std_training_data, Feature_scalers =
    standardize_features(encoded_training_data)

    Parameters = load_parameters("svr_both_rbf_shuffle.log")

    for Std_method in ["None", "Norm", "MinMax"]:  
        final_training_data, Label_scalers =
        label_std(std_training_data, Method=Std_method)  # standarize the
        labels/targets as well.
        
        # grid_search_tasks(std_training_data)
        # cv_tasks(std_training_data, 10, 32)
        # svr_training_test(final_training_data, Parameters, 
        Label_scalers=Label_scalers)

    def prepare_data(Datasheet, Parameter_file=None, 
        Label_std_method="MinMax"):
        """Prepare all data including scaling

        Parameters
        ------------
        Datasheet: str, full path to database spreadsheet file
        Parameters_file: str, full path to file that defines best
        parameters for different v.
        Label_std_method: str, label preprocessing method, one in ["None", "Norm", "MinMax"]
        Feature_std_method: str, feature preprocessing method, currently not used

"""
Training_data = read_spreadsheet("wild_and_mutant.csv")
Training_data = shuffle_data(Training_data)
Encoded_training_data, Encoders = one_hot_encode_features(Training_data)
Std_training_data, Feature_scalers = standardize_features(Encoded_training_data)

if Parameter_file != None:
    Parameters = load_parameters(Parameter_file)
else:
    Parameters = None

Final_training_data, Label_scalers = label_std(Std_training_data, Method=Label_std_method)  # standarize the labels/targets as well.

return Final_training_data, Feature_scalers, Label_scalers, Encoders, Parameters

if __name__ == "__main__":
    test_label_std()
    exit()

Datasheet = "wild_and_mutant.csv"
Parameter_file = "svr_both_rbf_shuffle.log"
Training_data, Feature_scalers, Label_scalers, Encoders, Parameters = prepare_data(Datasheet, Parameter_file=Parameter_file, Label_std_method="MinMax")

#    grid_search_tasks(std_training_data)
#    cv_tasks(Training_data, 10, 4, Label_scalers, Parameters)

#    reports = _validate_training_data(std_training_data)
#    for i, report in enumerate(reports, 1):
#        print("v = {}, duplicate data index = {}".format(i, report.values()))

models = train_model(Training_data, Parameters)
cPickle.dump(models, open("models_svm.p", "wb"))
cPickle.dump(Feature_scalers, open("feature_scalers.p", "wb"))
cPickle.dump(Encoders, open("encoders.p", "wb"))
cPickle.dump(Label_scalers, open("label_scalers.p", "wb"))

#    cPickle.dump(training_data, open("training_data.p", "wb"))
#    cPickle.dump(encoded_training_data, open("encoded_training_data.p", "wb"))
#    cPickle.dump(std_training_data, open("std_training_data.p", "wb"))
def quadprog_adjust(Substrates, Fluxes, Boundary_dict, Debug=False, Label_scalers=None):
    """adjust values from ML

    Parameters
    ==========
    Substrates: OrderedDict, keys as integers and values as floats, e.g., {1:0.25, 2:0, 3:0.75, ...}
    Fluxes: Dict, keys as integers and values as floats, e.g., {1:99.5, 2:1.1, ...}
    Debug: Boolean, True for showing debug info and False (default) for no.
    Label_scaler: sklearn.preprocessing.StandardScaler or .MinMaxScaler
    Forward transform is from fluxes in true range to scaled range
    Inverse transform is from scaled range to true range
    Boundary_dict: Upper boundaries and lower boundaries for 29 fluxes, depending on user inputs,
    e.g., {"lb29":999, "ub8":50}, populate ub and lb inequalities from them

    Returns
    =======
    Solution: Dict, keys as integers and values as floats, e.g., {1:99.5, 2:1.1, ...}

    Notes
    ======
    In Substrates, the mapping from keys to real chemicals is as follows:
    1. Glucose
    2. Fructose
    3. Galactose
    4. Gluconate
    5. Glutamate
    6. Citrate
    7. Xylose
    8. Succinate
    9. Malate
    10. Lactate
    11. Pyruvate
    12. Glycerol
    13. Acetate
    14. NaHCO3
Formulation of quadratic problems in MATLAB optimization toolbox are different from that in cvxopt. Here is a mapping between variables:

- \( H \Rightarrow P \) (the quadratic terms in objective function)
- \( f \Rightarrow q \) (the linear terms in objective function)
- \( A \) and \( \text{eyes for boundaries} \Rightarrow G \) (coefficients for linear terms in inequality constraints)
- \( b, -lb, ub \Rightarrow h \) (coefficients for constant terms in inequality constraints)
- \( A_{eq} \Rightarrow A \)
- \( Beq \Rightarrow b \)

Unimplemented features:
1. Using scaled values for quadprog

Example

```
>>> Substrates = {1:1, 2:0, 3:0, 4:0, 5:0, 6:0, 7:0, 8:0, 9:0, 10:0, 11:0, 12:0, 13:0, 14:0}
>>> Fluxes = Fluxes / 100  # turn it into column vector, 29x1
>>> import libflux
>>> libflux.quadprog_adjust(Substrates, Fluxes, {}, Debug=True)
>>> import cPickle
>>> Label_scalers = cPickle.load(open("label_scalers.p", "r"))
>>> libflux.quadprog_adjust(Substrates, Fluxes, {}, Debug=True, Label_scalers = Label_scalers)
>>> libflux.quadprog_adjust(Substrates, Fluxes, {"ub1":50}, Debug=True, Label_scalers = Label_scalers)
```

**Example**

```
import numpy
import cvxopt, cvxopt.solvers

Substrate2Index= {"glucose":1, "galactose":3, "fructose":2, "gluconate":4, "glutamate":5, "citrate":6, "xylose":7, "succinate":8, "malate":9, "lactate":10, "pyruvate":11, "glycerol":12, "acetate":13}

Ubs = numpy.array([[100,99.5,99.3,99.3,1216.6, 196.2,232,213.1,135,151.4, 113.7,94.1,41.2,47.5,71, 47.5,189,189,189,194, 194,194,181.5,55,148, 193.2,151,149.8,104.2043714]])
Ubs = Ubs.transpose()  # turn it into column vector, 29x1
```
Lbs = numpy.array([[0, -99.9, -51.5, -51.5, -13.5, -23.3, -36, -7.9, -144, 0, 0, -105, -106, -144.3, 0, 0, -100, -67.60986805, -13.5]])
Lbs = Lbs.transpose()  # turn it into column vector, 29x1

Aineq_bound, Bineq_bound = populate_boundary_inequalities(Boundary_dict)
Aineq = numpy.zeros((12+1, 29+1))  # the plus 1 is to tackle MATLAB 1-index
Aineq[1,1] = 1; Aineq[1,2] = -1; Aineq[1,10] = -1;
Aineq[2,2] = 1; Aineq[2,3] = -1; Aineq[2,15] = 1; Aineq[2,16] = 1;  #
Aineq[4,5] = 1; Aineq[4,6] = -1;
Aineq[5,6] = 1; Aineq[5,7] = -1; Aineq[5,28] = -1;
Aineq[6,7] = 1; Aineq[6,8] = -1; Aineq[6,25] = 1; Aineq[6,27] = -1;
Aineq[7,8] = 1; Aineq[7,9] = -1; Aineq[7,17] = -1; Aineq[7,24] = -1;
Aineq[8,13] = 1; Aineq[8,14] = -1;
Aineq[9,16] = 1; Aineq[9,15] = -1;
Aineq[10,19] = 1; Aineq[10,20] = -1;
Aineq[12,21] = -1; Aineq[12,22] = 1;
Aineq = Aineq[1:, 1:]  # convert 1-index to 0-index
Aineq = -1 * Aineq  # because in standarized formulation, it's Ax<=b but in our paper it is Ax>=b

if Label_scalers == None:  # if flux in their true range instead of scaled range
    Aineq = numpy.vstack([Aineq, -numpy.eye(29), numpy.eye(29)])
    # add eye matrixes for Lbs and Ubs

if not Aineq_bound == None:
    Aineq = numpy.vstack([Aineq, Aineq_bound])
else:
    Aineq = numpy.matrix(Aineq)

bineq = numpy.zeros((12+1, 1+1))
bineq[2,1] = 100 * Substrates[Substrate2Index["fructose"]]
bineq[6,1] = 100 * Substrates[Substrate2Index["pyruvate"]]
bineq[10,1] = 100 * Substrates[Substrate2Index["glutamate"]]
bineq = bineq[1:, 1:]  # convert 1-index to 0-index
# if Label_scalers == None: # if flux in their true range instead of scaled range
    bineq = numpy.vstack([bineq, -Lbs, Ubs])
if not Bineq_bound == None:
    bineq = numpy.vstack([bineq, Bineq_bound])
else:
    bineq = numpy.matrix(bineq)
Aeq = numpy.zeros((10+1, 29+1))
Aeq[1,1] = 1;
Aeq[2,3] = 1; Aeq[2,4] = -1;
Aeq[4,14] = 1; Aeq[4,16] = -1;
Aeq[6,18] = 1; Aeq[6,17] = -1;
Aeq[7,15] = 1; Aeq[7,12] = -1; Aeq[7,14] = 1;
Aeq[8,24] = 1; Aeq[8,18] = -1; Aeq[8,19] = 1;
Aeq[9,22] = -1; Aeq[9,23] = 1; Aeq[9,24] = -1; Aeq[9,29] = 1;
Aeq[10,20] = 1; Aeq[10,24] = 1; Aeq[10,21] = -1;
Aeq = Aeq[:, 1:] # convert 1-index to 0-index
Aeq = numpy.matrix(Aeq)
# Aeq = Aeq.transpose().tolist()
bmq = numpy.zeros((10+1,1+1))
bmq[1,1] = 100 * (Substrates[Substrate2Index["glucose"]]) +
                      Substrates[Substrate2Index["galactose"]])
bmq[2,1] = -100 * Substrates[Substrate2Index["glycerol"]]
bmq[5,1] = -100 * Substrates[Substrate2Index["glucanate"]]
bmq[6,1] = 100 * Substrates[Substrate2Index["citrate"]]
bmq[7,1] = 100 * Substrates[Substrate2Index["xylose"]]
bmq[9,1] = 100 * Substrates[Substrate2Index["malate"]]
bmq[10,1] = -100 * Substrates[Substrate2Index["succinate"]]
bmq = bmq[:, 1:] # convert 1-index to 0-index
bmq = numpy.matrix(bmq)
if Label_scalers == None:
    P = numpy.eye((29))
    q = [[Fluxes[i] for i in range(1, 29+1)]]
else: # convert non-scaled fluxes into [0,1]
    P = numpy.square(numpy.diag([Label_scalers[i].scale_ for i in
                        range(1, 29+1)]))
    q = [[Label_scalers[i].scale_**2 * Fluxes[i] for i in range(1,
                        29+1)]]
if Debug:
    # print P
    for i in range(1,29+1):
        pass
    q = -1*numpy.array((q)).transpose() # -1 is because -v_i but f.T*x
    in standard quadprog formalization
# print map(numpy.shape, [Aineq, bineq, Aeq, bmq, P, q])


```python
# print map(type, [Aineq, bineq, Aeq, beq, P, q])
# [bineq] = map(cvxopt.matrix, [bineq])
# [beq] = map(cvxopt.matrix, [beq])
# [Aineq, bineq, Aeq, beq] = map(cvxopt.matrix, [Aineq, bineq, Aeq, beq])

[Aineq, bineq, Aeq, beq, P, q] = map(cvxopt.matrix, [Aineq, bineq, Aeq, beq, P, q])

cvxopt.solvers.options['show_progress'] = False
Solv = cvxopt.solvers.qp(P, q, Aineq, bineq, Aeq, beq)
Solution = Solv['x']

Solution = numpy.array(Solution)[:,0] # conversion from cvxopt's matrix to numpy array

if Debug:
    numpy.set_printoptions(precision=4, suppress=True)
    print "<pre>"
    print ''.join([" V", " Adjusted ", " Predicted ", " Diff ", " Diff% ", " Diff%Rg "])
    for Idx, Value in enumerate(Solution):
        # print type((Ubs-Lbs)[Idx][0])
        Diff = Value - Fluxes[Idx+1]
        print "{0:2d}{1:10.3f}{2:10.3f}{3:10.3f}{4:8.1f}{5:8.1f}".format(Idx+1, Value, Fluxes[Idx+1], Diff, Diff/Fluxes[Idx+1]*100, Diff/((Ubs-Lbs)[Idx][0]*100) # convert from 0-index to 1-index
        print "</pre>"

    Solution = {i+1: Solution[i] for i in xrange(29)} # turn from numpy array to dict

return Solution

def test(S):
    print S

def print_influxes(Influxes):
    """Print influxes
    Influxes: dict, keys are influx id, values are floats
    """
```

import sys
sys.stderr = sys.stdout

# print Influxes
print ""
<h2>Influx values based on given parameters:</h2>
""
for ID, Value in Influxes.iteritems():
    print ""
    v%s = %.4f, <br>
    "" % (ID, Value)

# for x in range(5):
#     print x

for ID, Value in Influxes.iteritems():
    print ""
    v%s = %.4f, <br>
    "" % (ID, Value)

def populate_boundary_inequalities(Boundary_dict, Debug=False):
    ""
    Boundary_dict: Upper boundaries and lower boundaries for 29 fluxes,
    depending on user inputs,
    e.g., {"lb29":999, "ub8":50}, populate ub and lb
    inequalities from them

    Aineq: X-by-29 binary matrix, where N is the number of Ubs and Lbs
    set by user
    Bineq: X-by-1 column vector

    for any v_j <= p, there is Aineq[i][j] ==  1 and Bineq[j] ==  P
    for any v_j >= p, there is Aineq[i][j] == -1 and Bineq[j] == -p
    Note the inequalities are: Ax <= B
""
import numpy
if Boundary_dict == {}:
    return None, None

Row_vectors = []  # must be 29 columns and X rows where X is the number of Ubs and Lbs set by user
Boundary_column_vectors = []  # X rows and 1 column
for Polarity_Id, Bound_value in Boundary_dict.iteritems():
    Bound_type, Flux_ID = Polarity_Id[:2], int(Polarity_Id[2:])
    Row_vector = numpy.zeros(29)
    if Bound_type == "lb":
        Bound_value = -1*Bound_value
        Row_vector[Flux_ID-1] = -1.
    elif Bound_type == "ub":
    else:
        print "wrong boundary"
    Row_vectors.append(Row_vector)
    Boundary_column_vectors.append(Bound_value)
#        print "<br>", Bound_type, Flux_ID, Bound_value
Aineq = numpy.vstack(Row_vectors)
Bineq = numpy.vstack(Boundary_column_vectors)

if Debug:
    print "<pre>"
    print Aineq
    print Bineq
    print "</pre>"
return Aineq, Bineq

def process_boundaries(Form, Substrates):
    """Extract boundaries for fluxes from user input
    
    Form: cgi object
    Features: {}, empty dictionary by default
    
    Notes
    ======
    In Substrates, the mapping from keys to real chemicals is as follows:
    1. Glucose
    2. Fructose
    3. Galactose
    4. Gluconate
    5. Glutamate
    6. Citrate
    7. Xylose
    8. Succinate
    9. Malate
    """
import itertools
import cgi

Substrate2Index= {"glucose":1, "galactose":3, "fructose":2, "gluconate":4, "glutamate":5, "citrate":6, "xylose":7, "succinate":8, "malate":9, "lactate":10, "pyruvate":11, "glycerol":12, "acetate":13}
Feature_names = ["".join([Bound, ID]) for (Bound, ID) in itertools.product(["lb", "ub"], map(str, range(1, 29+1))) ]
Features= {}
for Feature_name in Feature_names:
    Feature_value = Form.getfirst(Feature_name)
    if Feature_value:
        Feature_value = cgi.escape(Feature_value)
        Features[Feature_name] = float(Feature_value) # convert all string to numbers
if Substrates[Substrate2Index["acetate"]]==0:
    Features["lb9"] = 0
if Substrates[Substrate2Index["lactate"]]==0:
    Features["lb27"] = 0

for Feature_name in Feature_names:
    print "%s is %s, " % (Feature_name, Features[Feature_name])
return Features

def process_input(Features):
    """Process the result from CGI parsing to form feature vector including substrate matrixi""
    Substrates: OrderedDict, keys as integers and values as floats
    1. Glucose
    2. Fructose
    3. Galactose
    4. Gluconate
    5. Glutamate
    6. Citrate
    7. Xylose
    8. Succinate
    9. Malate
    10. Lactate
    11. Pyruvate

10. Lactate
11. Pyruvate
12. Glycerol
13. Acetate
14. NaHCO3
12. Glycerol
13. Acetate
14. NaHCO₃

Feature vectors order: [Species, Reactor, Nutrient, Oxygen, Method, MFA, Energy, Growth_rate, Substrate_uptake_rate] + ratio of 14 carbon sources in the order above

```
Num_substrates = 14 # excluding other carbon
# Generate substrate matrix
import collections
Substrates = collections.OrderedDict([(i,0) for i in range(1, Num_substrates+1)]) # substrate values, initialization
Substrates[int(Features["Substrate_first"]) ] += Features["Ratio_first"]
Substrates[int(Features["Substrate_sec"]) ] += Features["Ratio_sec"]

# Form the feature vector
Vector += [Substrates[i] for i in range(1, Num_substrates+1)]
Vector.append(Features["Substrate_other"] ) # Other carbon source

# Print input check
import clp
DB = clp.process_species_db("SI_1_species_db.csv")
P = clp.species_db_to_constraints(DB)
if not clp.input_ok(P, Vector):
    print "<p><font color="red">The input data might violate the oxygen, substrate uptake rate or carbon sources of the selected species. Therefore, the following prediction may not be biologically meaningful. Please check your inputs!</font></p>"

# Print debug info

Substrate_dict = collections.OrderedDict([(i+1,Name) for i, Name in enumerate(Substrate_names)])
print "<p>Feature Vector (pre-one-hot-encoding and pre-scaling):", Vector, "</br>"
print "in which the substrates ratios are:", [(Substrate_dict[Index],Ratio) for Index, Ratio in Substrates.items()]
print "<br>Feature vector size is ", len(Vector), "</p>"

return Vector, Substrates
```
def rule_adjust(Influxes, Substrates):
    """Adjust influxes values using rules
    """

    Substrate2Index= {
        "glucose":1, "galactose":3, "fructose":2, 
        "gluconate":4, "glutamate":5, "citrate":6, "xylose":7, "succinate":8, 
        "malate":9, "lactate":10, "pyruvate":11, "glycerol":12, "acetate":13
    }

    #Step 1: Compute dependent influxes
    # Influxes[1] = 100 * Substrates[Substrate2Index["glucose"]]
    # Influxes[16] = Influxes[14]
    # Influxes[18] = Influxes[17] + 100 * Substrates[Substrate2Index["citrate"]]
    # Influxes[22] = Influxes[21]
    # Influxes[29] = Influxes[22] + Influxes[24] - Influxes[23] + 100 * Substrates[Substrate2Index["malate"]]
    
    # Step 2: Correct flux values
    if Substrates[Substrate2Index["acetate"]] != 0:
        Influxes[9] = -100 * Substrates[Substrate2Index["acetate"]]
    if Substrates[Substrate2Index["lactate"]] != 0:
        Influxes[27] = -100 * Substrates[Substrate2Index["lactate"]]

    return Influxes

def predict(Vector, Substrates, Boundary_dict):
    """Predict and adjust all influx values
    """

    Vector: 1-D list of floats, the feature vector, including substrate matrix, size = 24
    Substrates: dict of floats, 1-indexed part of Feature_vector, ratio of substrates
    Boundary_dict: Upper boundaries and lower boundaries for 29 fluxes, depending on user inputs,
    e.g., {"lb29":999, "ub8":50}, populate ub and lb inequalities from them.
    If no boundary set by user, it can be an empty dictionary

    Calls adjust_influxes() to compute dependent influxes.
    """

    import cPickle
    import time
import collections
import sys

Models = cPickle.load(open("models_svm.p", "r"))
Feature_scalers = cPickle.load(open("feature_scalers.p", "r"))
Encoders = cPickle.load(open("encoders.p", "r"))
Label_scalers = cPickle.load(open("label_scalers.p", "r"))

print "<p>Models, feature and label Scalers and one-hot Encoder loaded..</p>"

T = time.clock()
Influxes = {}

for vID, Model in Models.iteritems():
    Vector_local = list(Vector)  # make a copy; o/w Vector will be changed in one-hot encoding and standardization for different models
    Vector_local = Vector[:6+1:]  # combine one-hot-encoded categorical features with continuous features (including substrate matrix)
    Vector_local = Feature_scalers[vID].transform(Vector_local)  # standardization of features
    Influx_local = Model.predict(Vector_local)[0]  # prediction
    Influxes[vID] = Influx_local

Influxes = quadprog_adjust(Substrates, Influxes, Boundary_dict, Label_scalers=Label_scalers, Debug=True)
Influxes = rule_adjust(Influxes, Substrates)

T = time.clock() - T

print_influxes(Influxes)

print """"</p>\
<p>Using RBF kernel SVM as regressor. Parameters vary for different fluxes. For details, refer to <a href="svr_both_rbf_shuffle.log">this document generated by grid search on SVM parameters</a>. </p><p>Standardization and Regression done in %s seconds.</p>

```
return Influxes
```

---

S2: Detailed information of case study (20 cases)

Summary of 20 cases

<table>
<thead>
<tr>
<th>Case number</th>
<th>Root mean squared error (RMSE)</th>
<th>Average flux</th>
<th>ML</th>
<th>MFlux</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>27.1</td>
<td>6.9</td>
<td>6.5</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>23.9</td>
<td>17.5</td>
<td>10.6</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>22.8</td>
<td>16.8</td>
<td>8.2</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>22.1</td>
<td>16.5</td>
<td>7.6</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>20.9</td>
<td>7.0</td>
<td>6.9</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>22.3</td>
<td>8.8</td>
<td>8.2</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>27.2</td>
<td>5.9</td>
<td>5.3</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>58.5</td>
<td>4.5</td>
<td>3.2</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>---</td>
<td>-----</td>
<td>-----</td>
<td>-----</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>36.1</td>
<td>4.2</td>
<td>3.2</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>55.8</td>
<td>4.2</td>
<td>3.3</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>24.0</td>
<td>5.9</td>
<td>4.3</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>14.7</td>
<td>9.4</td>
<td>8.8</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>15.8</td>
<td>11.7</td>
<td>10.1</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>27.1</td>
<td>4.1</td>
<td>3.4</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>21.6</td>
<td>2.9</td>
<td>3.6</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>34.4</td>
<td>4.0</td>
<td>3.0</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>46.7</td>
<td>6.5</td>
<td>4.1</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>46.6</td>
<td>6.6</td>
<td>4.3</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>62.1</td>
<td>5.1</td>
<td>4.0</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>60.2</td>
<td>5.1</td>
<td>3.6</td>
<td></td>
</tr>
<tr>
<td>average</td>
<td>33.5</td>
<td>7.7</td>
<td>5.6</td>
<td></td>
</tr>
</tbody>
</table>

**Case 1**

Case 1 heat map

Case 1 results

<table>
<thead>
<tr>
<th>13C flux</th>
<th>ML</th>
<th>MFlux</th>
</tr>
</thead>
<tbody>
<tr>
<td>v1 EMP</td>
<td>100.0</td>
<td>100.0</td>
</tr>
<tr>
<td>V2 EMP</td>
<td>71.6</td>
<td>73.3</td>
</tr>
<tr>
<td>V3 EMP</td>
<td>82.6</td>
<td>80.0</td>
</tr>
<tr>
<td>V4 EMP</td>
<td>82.6</td>
<td>79.9</td>
</tr>
<tr>
<td>V5 EMP</td>
<td>170.5</td>
<td>175.8</td>
</tr>
<tr>
<td>V6 EMP</td>
<td>157.6</td>
<td>164.6</td>
</tr>
<tr>
<td>V7 EMP</td>
<td>127.1</td>
<td>130.1</td>
</tr>
<tr>
<td>V8</td>
<td>114.0</td>
<td>116.6</td>
</tr>
<tr>
<td>V9</td>
<td>66.8</td>
<td>73.8</td>
</tr>
<tr>
<td>V10</td>
<td>26.8</td>
<td>26.8</td>
</tr>
<tr>
<td>V11</td>
<td>25.4</td>
<td>17.2</td>
</tr>
<tr>
<td>V12</td>
<td>11.5</td>
<td>8.7</td>
</tr>
<tr>
<td>V13</td>
<td>13.9</td>
<td>11.5</td>
</tr>
<tr>
<td>V14</td>
<td>7.1</td>
<td>5.8</td>
</tr>
<tr>
<td>V15</td>
<td>4.4</td>
<td>2.5</td>
</tr>
<tr>
<td>V16</td>
<td>7.1</td>
<td>5.7</td>
</tr>
<tr>
<td>V17 TCA cycle</td>
<td>19.0</td>
<td>25.6</td>
</tr>
<tr>
<td>V18 TCA cycle</td>
<td>19.0</td>
<td>41.1</td>
</tr>
<tr>
<td>V19 TCA cycle</td>
<td>16.6</td>
<td>30.5</td>
</tr>
<tr>
<td>V20 TCA cycle</td>
<td>4.9</td>
<td>19.3</td>
</tr>
<tr>
<td>V21 TCA cycle</td>
<td>10.9</td>
<td>19.0</td>
</tr>
<tr>
<td>V22 TCA cycle</td>
<td>13.7</td>
<td>18.5</td>
</tr>
<tr>
<td>V23 TCA cycle</td>
<td>10.9</td>
<td>12.8</td>
</tr>
<tr>
<td>V24 Glyoxylate</td>
<td>2.5</td>
<td>4.0</td>
</tr>
<tr>
<td>V25 ED</td>
<td>1.4</td>
<td>1.5</td>
</tr>
<tr>
<td>V26 ETOH</td>
<td>0.0</td>
<td>0.1</td>
</tr>
<tr>
<td>V27 LAC</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>V28</td>
<td>24.7</td>
<td>15.5</td>
</tr>
<tr>
<td>V29</td>
<td>5.3</td>
<td>1.4</td>
</tr>
<tr>
<td>RMSE</td>
<td>6.9</td>
<td>6.5</td>
</tr>
</tbody>
</table>

**Case 2-4**

Case 2 heat map
Case 3 heat map
Case 4 heat map
### Case 2 - 4 results

<table>
<thead>
<tr>
<th></th>
<th>13C-flux</th>
<th>ML</th>
<th>MFlux</th>
<th>13C-flux</th>
<th>ML</th>
<th>MFlux</th>
<th>13C-flux</th>
<th>ML</th>
<th>MFlux</th>
</tr>
</thead>
<tbody>
<tr>
<td>v1 EMP</td>
<td>100.0</td>
<td>100.2</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
</tr>
<tr>
<td>v2 EMP</td>
<td>62.0</td>
<td>65.0</td>
<td>61.0</td>
<td>61.0</td>
<td>67.6</td>
<td>67.6</td>
<td>61.0</td>
<td>67.8</td>
<td>67.8</td>
</tr>
<tr>
<td>v3 EMP</td>
<td>76.7</td>
<td>79.1</td>
<td>79.1</td>
<td>76.3</td>
<td>79.0</td>
<td>79.4</td>
<td>74.7</td>
<td>79.2</td>
<td>79.6</td>
</tr>
<tr>
<td>v4 EMP</td>
<td>76.7</td>
<td>79.2</td>
<td>79.1</td>
<td>76.3</td>
<td>79.8</td>
<td>79.4</td>
<td>74.7</td>
<td>80.0</td>
<td>79.6</td>
</tr>
<tr>
<td>v5 EMP</td>
<td>166.7</td>
<td>161.7</td>
<td>161.7</td>
<td>165.0</td>
<td>167.0</td>
<td>167.0</td>
<td>162.7</td>
<td>166.6</td>
<td>166.6</td>
</tr>
<tr>
<td>v6 EMP</td>
<td>156.0</td>
<td>142.0</td>
<td>142.0</td>
<td>151.0</td>
<td>156.1</td>
<td>156.1</td>
<td>149.0</td>
<td>155.4</td>
<td>155.4</td>
</tr>
<tr>
<td>v7 EMP</td>
<td>156.0</td>
<td>125.4</td>
<td>125.4</td>
<td>150.0</td>
<td>122.4</td>
<td>122.4</td>
<td>147.0</td>
<td>121.1</td>
<td>121.1</td>
</tr>
<tr>
<td>v8</td>
<td>149.0</td>
<td>118.7</td>
<td>139.5</td>
<td>144.0</td>
<td>117.8</td>
<td>128.1</td>
<td>147.0</td>
<td>116.3</td>
<td>126.0</td>
</tr>
<tr>
<td>v9</td>
<td>37.0</td>
<td>31.2</td>
<td>30.0</td>
<td>35.0</td>
<td>27.4</td>
<td>38.0</td>
<td>28.9</td>
<td>27.3</td>
<td></td>
</tr>
<tr>
<td>v10</td>
<td>38.0</td>
<td>35.1</td>
<td>39.0</td>
<td>29.0</td>
<td>30.0</td>
<td>38.0</td>
<td>28.9</td>
<td>27.3</td>
<td></td>
</tr>
<tr>
<td>v11</td>
<td>30.0</td>
<td>53.0</td>
<td>39.0</td>
<td>31.0</td>
<td>26.7</td>
<td>29.0</td>
<td>26.2</td>
<td>25.0</td>
<td></td>
</tr>
<tr>
<td>v12</td>
<td>16.7</td>
<td>18.9</td>
<td>18.2</td>
<td>15.0</td>
<td>16.1</td>
<td>12.8</td>
<td>13.3</td>
<td>16.1</td>
<td>12.7</td>
</tr>
<tr>
<td>v13</td>
<td>13.3</td>
<td>5.9</td>
<td>20.8</td>
<td>16.3</td>
<td>8.7</td>
<td>6.4</td>
<td>9.1</td>
<td>12.3</td>
<td></td>
</tr>
<tr>
<td>v14</td>
<td>10.0</td>
<td>7.5</td>
<td>9.1</td>
<td>8.8</td>
<td>5.9</td>
<td>8.0</td>
<td>5.9</td>
<td>6.4</td>
<td></td>
</tr>
<tr>
<td>v15</td>
<td>6.7</td>
<td>7.0</td>
<td>6.3</td>
<td>5.3</td>
<td>6.4</td>
<td>5.3</td>
<td>6.4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>v16</td>
<td>10.0</td>
<td>7.4</td>
<td>9.1</td>
<td>8.8</td>
<td>5.3</td>
<td>8.0</td>
<td>5.3</td>
<td>6.4</td>
<td></td>
</tr>
<tr>
<td>v17 TCA cycle</td>
<td>53.0</td>
<td>61.5</td>
<td>69.5</td>
<td>51.0</td>
<td>48.8</td>
<td>58.1</td>
<td>54.0</td>
<td>47.9</td>
<td>56.0</td>
</tr>
<tr>
<td>v18 TCA cycle</td>
<td>53.0</td>
<td>78.1</td>
<td>69.5</td>
<td>51.0</td>
<td>53.0</td>
<td>58.1</td>
<td>54.0</td>
<td>49.4</td>
<td>56.0</td>
</tr>
<tr>
<td>v19 TCA cycle</td>
<td>11.0</td>
<td>44.1</td>
<td>29.5</td>
<td>11.0</td>
<td>40.0</td>
<td>18.1</td>
<td>11.0</td>
<td>37.8</td>
<td>16.0</td>
</tr>
<tr>
<td>v20 TCA cycle</td>
<td>0.0</td>
<td>39.3</td>
<td>17.0</td>
<td>0.0</td>
<td>34.0</td>
<td>9.9</td>
<td>0.0</td>
<td>33.0</td>
<td>8.9</td>
</tr>
<tr>
<td>v21 TCA cycle</td>
<td>42.0</td>
<td>49.5</td>
<td>57.0</td>
<td>40.0</td>
<td>32.3</td>
<td>49.9</td>
<td>43.0</td>
<td>31.4</td>
<td>48.9</td>
</tr>
<tr>
<td>v22 TCA cycle</td>
<td>42.0</td>
<td>46.3</td>
<td>57.0</td>
<td>40.0</td>
<td>49.9</td>
<td>43.0</td>
<td>49.1</td>
<td>48.9</td>
<td></td>
</tr>
<tr>
<td>v23 TCA cycle</td>
<td>73.0</td>
<td>40.9</td>
<td>79.0</td>
<td>70.0</td>
<td>25.8</td>
<td>69.4</td>
<td>67.0</td>
<td>25.3</td>
<td>69.1</td>
</tr>
<tr>
<td>v24 Glyoxylate</td>
<td>42.0</td>
<td>7.2</td>
<td>40.0</td>
<td>40.0</td>
<td>5.4</td>
<td>40.0</td>
<td>43.0</td>
<td>5.6</td>
<td>40.0</td>
</tr>
<tr>
<td>v25 ED</td>
<td>8.0</td>
<td>1.7</td>
<td>0.0</td>
<td>7.0</td>
<td>0.9</td>
<td>2.4</td>
<td>9.0</td>
<td>0.7</td>
<td>2.3</td>
</tr>
<tr>
<td>v26 ETOH</td>
<td>0.0</td>
<td>-0.1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.2</td>
<td>0.0</td>
<td>0.0</td>
<td>0.3</td>
<td>0.0</td>
</tr>
<tr>
<td>v27 LAC</td>
<td>0.0</td>
<td>-0.1</td>
<td>-0.1</td>
<td>0.0</td>
<td>-0.1</td>
<td>-0.1</td>
<td>-0.1</td>
<td>-0.1</td>
<td>-0.1</td>
</tr>
<tr>
<td>v28</td>
<td>-7.0</td>
<td>14.3</td>
<td>0.0</td>
<td>-6.0</td>
<td>26.3</td>
<td>0.0</td>
<td>0.0</td>
<td>25.6</td>
<td>0.0</td>
</tr>
<tr>
<td>v29</td>
<td>11.0</td>
<td>13.0</td>
<td>18.0</td>
<td>10.0</td>
<td>14.7</td>
<td>20.4</td>
<td>19.0</td>
<td>14.1</td>
<td>19.8</td>
</tr>
</tbody>
</table>

**RMSE** | 17.5 | 10.6 | 16.8 | 8.2 | 16.5 | 7.6

### Case 5-7

241

Case 5 heat map
Case 6 heat map
Case 7 heat map
<table>
<thead>
<tr>
<th></th>
<th>case 5</th>
<th></th>
<th>case 6</th>
<th></th>
<th>case 7</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>13C-flux</td>
<td>ML</td>
<td>MFlux</td>
<td>13C-flux</td>
<td>ML</td>
</tr>
<tr>
<td>v1 EMP</td>
<td>100.0</td>
<td>99.4</td>
<td>100.0</td>
<td>100.0</td>
<td>99.0</td>
</tr>
<tr>
<td>v2 EMP</td>
<td>66.3</td>
<td>63.1</td>
<td>66.8</td>
<td>60.6</td>
<td>66.4</td>
</tr>
<tr>
<td>v3 EMP</td>
<td>82.6</td>
<td>81.1</td>
<td>83.6</td>
<td>79.2</td>
<td>80.0</td>
</tr>
<tr>
<td>v4 EMP</td>
<td>82.6</td>
<td>81.0</td>
<td>83.6</td>
<td>80.2</td>
<td>80.0</td>
</tr>
<tr>
<td>v5 EMP</td>
<td>170.9</td>
<td>170.2</td>
<td>173.4</td>
<td>172.4</td>
<td>171.7</td>
</tr>
<tr>
<td>v6 EMP</td>
<td>158.8</td>
<td>158.5</td>
<td>158.5</td>
<td>162.6</td>
<td>167.1</td>
</tr>
<tr>
<td>v7 EMP</td>
<td>162.0</td>
<td>130.2</td>
<td>130.2</td>
<td>159.2</td>
<td>128.8</td>
</tr>
<tr>
<td>v8</td>
<td>103.3</td>
<td>94.6</td>
<td>94.6</td>
<td>113.1</td>
<td>107.2</td>
</tr>
<tr>
<td>v9</td>
<td>45.8</td>
<td>43.2</td>
<td>43.2</td>
<td>62.1</td>
<td>47.6</td>
</tr>
<tr>
<td>v10</td>
<td>31.9</td>
<td>34.2</td>
<td>36.2</td>
<td>31.8</td>
<td>37.3</td>
</tr>
<tr>
<td>v11</td>
<td>31.9</td>
<td>35.7</td>
<td>36.2</td>
<td>31.8</td>
<td>42.6</td>
</tr>
<tr>
<td>v12</td>
<td>16.3</td>
<td>20.4</td>
<td>19.7</td>
<td>16.7</td>
<td>20.2</td>
</tr>
<tr>
<td>v13</td>
<td>15.6</td>
<td>18.8</td>
<td>16.5</td>
<td>15.0</td>
<td>24.0</td>
</tr>
<tr>
<td>v14</td>
<td>10.0</td>
<td>11.0</td>
<td>11.4</td>
<td>10.0</td>
<td>10.5</td>
</tr>
<tr>
<td>v15</td>
<td>6.4</td>
<td>8.9</td>
<td>8.4</td>
<td>6.8</td>
<td>8.7</td>
</tr>
<tr>
<td>v16</td>
<td>10.0</td>
<td>11.9</td>
<td>11.4</td>
<td>10.0</td>
<td>13.2</td>
</tr>
<tr>
<td>v17 TCA cycle</td>
<td>49.5</td>
<td>42.5</td>
<td>47.8</td>
<td>44.3</td>
<td>45.9</td>
</tr>
<tr>
<td>v18 TCA cycle</td>
<td>49.5</td>
<td>49.4</td>
<td>47.8</td>
<td>44.3</td>
<td>36.8</td>
</tr>
<tr>
<td>v19 TCA cycle</td>
<td>49.5</td>
<td>51.5</td>
<td>47.8</td>
<td>44.3</td>
<td>59.7</td>
</tr>
<tr>
<td>v20 TCA cycle</td>
<td>39.8</td>
<td>31.1</td>
<td>31.2</td>
<td>35.7</td>
<td>38.6</td>
</tr>
<tr>
<td>v21 TCA cycle</td>
<td>39.8</td>
<td>31.6</td>
<td>31.2</td>
<td>35.7</td>
<td>33.5</td>
</tr>
<tr>
<td>v22 TCA cycle</td>
<td>39.8</td>
<td>33.0</td>
<td>35.6</td>
<td>35.7</td>
<td>50.5</td>
</tr>
<tr>
<td>v23 TCA cycle</td>
<td>31.8</td>
<td>29.2</td>
<td>26.2</td>
<td>19.2</td>
<td>26.1</td>
</tr>
<tr>
<td>v24 Glyoxylate</td>
<td>0.0</td>
<td>-1.7</td>
<td>0.0</td>
<td>0.0</td>
<td>-2.8</td>
</tr>
<tr>
<td>v25 ED</td>
<td>0.0</td>
<td>-1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>-0.6</td>
</tr>
<tr>
<td>v26 ETOH</td>
<td>0.0</td>
<td>0.0</td>
<td>0.1</td>
<td>0.0</td>
<td>-0.2</td>
</tr>
<tr>
<td>v27 LAC</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.1</td>
</tr>
<tr>
<td>v28</td>
<td>33.0</td>
<td>30.9</td>
<td>38.8</td>
<td>27.9</td>
<td>27.9</td>
</tr>
<tr>
<td>v29</td>
<td>8.1</td>
<td>9.8</td>
<td>9.4</td>
<td>16.5</td>
<td>13.3</td>
</tr>
<tr>
<td>RMSE</td>
<td>7.0</td>
<td>6.9</td>
<td>8.8</td>
<td>8.2</td>
<td>5.9</td>
</tr>
</tbody>
</table>

245
Case 8 – 11


<table>
<thead>
<tr>
<th></th>
<th>succinate + glutamate</th>
<th>glycerol</th>
<th>Malate</th>
<th>Fructose</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>case 8</td>
<td>case 9</td>
<td>case 10</td>
<td>case 11</td>
</tr>
<tr>
<td>13C-flux</td>
<td>ML</td>
<td>MFlux</td>
<td>ML</td>
<td>MFlux</td>
</tr>
<tr>
<td>v1 EMP</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>v2 EMP</td>
<td>-7.1</td>
<td>-5.0</td>
<td>-8.2</td>
<td>-2.4</td>
</tr>
<tr>
<td>v3 EMP</td>
<td>-6.5</td>
<td>-5.0</td>
<td>-4.3</td>
<td>-5.1</td>
</tr>
<tr>
<td>v4 EMP</td>
<td>-6.5</td>
<td>-6.3</td>
<td>-4.3</td>
<td>94.9</td>
</tr>
<tr>
<td>v5 EMP</td>
<td>0.0</td>
<td>2.2</td>
<td>-6.5</td>
<td>76.9</td>
</tr>
<tr>
<td>v6 EMP</td>
<td>-23.3</td>
<td>-21.0</td>
<td>-21.0</td>
<td>76.9</td>
</tr>
<tr>
<td>v7 EMP</td>
<td>45.8</td>
<td>46.0</td>
<td>46.0</td>
<td>72.2</td>
</tr>
<tr>
<td>v8</td>
<td>44.0</td>
<td>41.9</td>
<td>41.9</td>
<td>41.7</td>
</tr>
<tr>
<td>v9</td>
<td>0.0</td>
<td>-2.9</td>
<td>-2.9</td>
<td>1.8</td>
</tr>
<tr>
<td>v10</td>
<td>4.2</td>
<td>4.2</td>
<td>8.2</td>
<td>0.1</td>
</tr>
<tr>
<td>v11</td>
<td>4.2</td>
<td>15.6</td>
<td>8.2</td>
<td>0.1</td>
</tr>
<tr>
<td>v12</td>
<td>0.6</td>
<td>1.9</td>
<td>2.6</td>
<td>-2.6</td>
</tr>
<tr>
<td>v13</td>
<td>3.6</td>
<td>3.7</td>
<td>5.7</td>
<td>2.8</td>
</tr>
<tr>
<td>v14</td>
<td>1.2</td>
<td>0.7</td>
<td>2.0</td>
<td>-0.2</td>
</tr>
<tr>
<td>v15</td>
<td>-0.6</td>
<td>0.2</td>
<td>0.6</td>
<td>-2.4</td>
</tr>
<tr>
<td>v16</td>
<td>1.2</td>
<td>1.7</td>
<td>2.0</td>
<td>-0.2</td>
</tr>
<tr>
<td>v17 TCA cycle</td>
<td>35.3</td>
<td>35.3</td>
<td>42.8</td>
<td>32.2</td>
</tr>
<tr>
<td>v18 TCA cycle</td>
<td>35.3</td>
<td>37.2</td>
<td>42.8</td>
<td>32.2</td>
</tr>
<tr>
<td>v19 TCA cycle</td>
<td>35.3</td>
<td>54.2</td>
<td>41.2</td>
<td>32.2</td>
</tr>
<tr>
<td>v20 TCA cycle</td>
<td>69.3</td>
<td>67.3</td>
<td>65.5</td>
<td>26.0</td>
</tr>
<tr>
<td>v21 TCA cycle</td>
<td>129.5</td>
<td>126.6</td>
<td>127.4</td>
<td>26.0</td>
</tr>
<tr>
<td>v22 TCA cycle</td>
<td>129.5</td>
<td>126.5</td>
<td>127.4</td>
<td>26.0</td>
</tr>
<tr>
<td>v23 TCA cycle</td>
<td>111.4</td>
<td>108.1</td>
<td>110.6</td>
<td>10.0</td>
</tr>
<tr>
<td>v24 Glyoxylate</td>
<td>0.0</td>
<td>3.4</td>
<td>1.6</td>
<td>0.0</td>
</tr>
<tr>
<td>v25 ED</td>
<td>0.0</td>
<td>-1.5</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>v26 ETOH</td>
<td>0.0</td>
<td>0.1</td>
<td>0.1</td>
<td>0.0</td>
</tr>
<tr>
<td>v27 LAC</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>v28</td>
<td>-67.6</td>
<td>-65.5</td>
<td>-65.5</td>
<td>31.6</td>
</tr>
</tbody>
</table>

246
Case 9 heat map
Case 10 heat map
Case 11 heat map
Case 12


<table>
<thead>
<tr>
<th>prefix</th>
<th>case 12</th>
<th>13C-flux</th>
<th>ML</th>
<th>MFlux</th>
</tr>
</thead>
<tbody>
<tr>
<td>v1 EMP</td>
<td></td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
</tr>
<tr>
<td>v2 EMP</td>
<td></td>
<td>66.9</td>
<td>45.3</td>
<td>45.3</td>
</tr>
<tr>
<td>v3 EMP</td>
<td></td>
<td>81.4</td>
<td>76.3</td>
<td>76.4</td>
</tr>
<tr>
<td>v4 EMP</td>
<td></td>
<td>81.4</td>
<td>76.5</td>
<td>76.4</td>
</tr>
<tr>
<td>v5 EMP</td>
<td></td>
<td>168.5</td>
<td>166.1</td>
<td>166.1</td>
</tr>
<tr>
<td>v6 EMP</td>
<td></td>
<td>155.6</td>
<td>153.4</td>
<td>153.4</td>
</tr>
<tr>
<td>v7 EMP</td>
<td></td>
<td>150.4</td>
<td>147.6</td>
<td>147.6</td>
</tr>
<tr>
<td>v8</td>
<td></td>
<td>89.4</td>
<td>78.4</td>
<td>78.4</td>
</tr>
<tr>
<td>v9</td>
<td></td>
<td>0.0</td>
<td>0.9</td>
<td>0.9</td>
</tr>
<tr>
<td>v10</td>
<td></td>
<td>31.3</td>
<td>57.9</td>
<td>50.5</td>
</tr>
<tr>
<td>v11</td>
<td></td>
<td>31.3</td>
<td>44.8</td>
<td>50.5</td>
</tr>
<tr>
<td>v12</td>
<td></td>
<td>15.1</td>
<td>27.9</td>
<td>27.9</td>
</tr>
<tr>
<td>v13</td>
<td></td>
<td>16.3</td>
<td>24.9</td>
<td>22.6</td>
</tr>
<tr>
<td>v14</td>
<td></td>
<td>8.7</td>
<td>14.6</td>
<td>16.0</td>
</tr>
<tr>
<td>v15</td>
<td></td>
<td>6.4</td>
<td>12.6</td>
<td>11.9</td>
</tr>
<tr>
<td>v16</td>
<td></td>
<td>8.7</td>
<td>17.7</td>
<td>16.0</td>
</tr>
<tr>
<td>v17 TCA cycle</td>
<td></td>
<td>64.0</td>
<td>58.6</td>
<td>55.3</td>
</tr>
<tr>
<td>v18 TCA cycle</td>
<td></td>
<td>64.0</td>
<td>62.2</td>
<td>55.3</td>
</tr>
<tr>
<td>v19 TCA cycle</td>
<td></td>
<td>64.0</td>
<td>45.0</td>
<td>55.3</td>
</tr>
<tr>
<td>v20 TCA cycle</td>
<td></td>
<td>53.3</td>
<td>51.2</td>
<td>50.2</td>
</tr>
<tr>
<td>v21 TCA cycle</td>
<td></td>
<td>43.1</td>
<td>47.8</td>
<td>50.2</td>
</tr>
<tr>
<td>v22 TCA cycle</td>
<td></td>
<td>55.5</td>
<td>52.4</td>
<td>55.6</td>
</tr>
<tr>
<td>v23 TCA cycle</td>
<td></td>
<td>64.1</td>
<td>55.7</td>
<td>51.9</td>
</tr>
<tr>
<td>v24 Glyoxylate</td>
<td></td>
<td>0.0</td>
<td>5.0</td>
<td>0.0</td>
</tr>
<tr>
<td>v25 ED</td>
<td></td>
<td>0.0</td>
<td>0.5</td>
<td>0.0</td>
</tr>
<tr>
<td>v26 ETOH</td>
<td></td>
<td>0.0</td>
<td>0.0</td>
<td>0.1</td>
</tr>
<tr>
<td>v27 LAC</td>
<td></td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>v28</td>
<td></td>
<td>32.7</td>
<td>30.8</td>
<td>30.8</td>
</tr>
<tr>
<td>v29</td>
<td></td>
<td>N.A.</td>
<td>4.2</td>
<td>3.7</td>
</tr>
</tbody>
</table>

RMSE: 9.4 8.8
Case 12 heat map

Case 13

<table>
<thead>
<tr>
<th>case 13</th>
<th>13C-flux</th>
<th>ML</th>
<th>MFlux</th>
</tr>
</thead>
<tbody>
<tr>
<td>v1 EMP</td>
<td>100</td>
<td>99.7</td>
<td>100.0</td>
</tr>
<tr>
<td>v2 EMP</td>
<td>40.2</td>
<td>43.8</td>
<td>43.1</td>
</tr>
<tr>
<td>v3 EMP</td>
<td>70.9</td>
<td>75.3</td>
<td>75.4</td>
</tr>
<tr>
<td>v4 EMP</td>
<td>70.9</td>
<td>75.5</td>
<td>75.4</td>
</tr>
<tr>
<td>v5 EMP</td>
<td>155.9</td>
<td>169.3</td>
<td>169.3</td>
</tr>
<tr>
<td>v6 EMP</td>
<td>147.6</td>
<td>164.3</td>
<td>164.3</td>
</tr>
<tr>
<td>v7 EMP</td>
<td>147.8</td>
<td>165.0</td>
<td>165.0</td>
</tr>
<tr>
<td>v8</td>
<td>70.6</td>
<td>84.3</td>
<td>84.3</td>
</tr>
<tr>
<td>v9</td>
<td>1.1</td>
<td>6.6</td>
<td>6.6</td>
</tr>
<tr>
<td>v10</td>
<td>55.1</td>
<td>58.9</td>
<td>56.9</td>
</tr>
<tr>
<td>v11</td>
<td>55.1</td>
<td>53.0</td>
<td>53.7</td>
</tr>
<tr>
<td>v12</td>
<td>32.5</td>
<td>28.5</td>
<td>29.1</td>
</tr>
<tr>
<td>v13</td>
<td>22.6</td>
<td>24.3</td>
<td>24.6</td>
</tr>
<tr>
<td>v14</td>
<td>17.8</td>
<td>14.7</td>
<td>16.3</td>
</tr>
<tr>
<td>v15</td>
<td>14.7</td>
<td>13.0</td>
<td>12.9</td>
</tr>
<tr>
<td>v16</td>
<td>17.8</td>
<td>17.9</td>
<td>16.3</td>
</tr>
<tr>
<td>v17 TCA cycle</td>
<td>50.2</td>
<td>65.4</td>
<td>63.9</td>
</tr>
<tr>
<td>v18 TCA cycle</td>
<td>50.2</td>
<td>66.6</td>
<td>63.9</td>
</tr>
<tr>
<td>v19 TCA cycle</td>
<td>50.2</td>
<td>54.9</td>
<td>59.2</td>
</tr>
<tr>
<td>v20 TCA cycle</td>
<td>42.1</td>
<td>59.1</td>
<td>57.2</td>
</tr>
<tr>
<td>v21 TCA cycle</td>
<td>42.1</td>
<td>58.2</td>
<td>61.9</td>
</tr>
<tr>
<td>v22 TCA cycle</td>
<td>42.1</td>
<td>59.6</td>
<td>61.9</td>
</tr>
<tr>
<td>v23 TCA cycle</td>
<td>42.1</td>
<td>61.3</td>
<td>59.4</td>
</tr>
<tr>
<td>v24 Glyoxylate</td>
<td>0</td>
<td>4.3</td>
<td>4.7</td>
</tr>
<tr>
<td>v25 ED</td>
<td>0</td>
<td>1.6</td>
<td>3.2</td>
</tr>
<tr>
<td>v26 ETOH</td>
<td>0</td>
<td>0.0</td>
<td>0.1</td>
</tr>
<tr>
<td>v27 LAC</td>
<td>2.2</td>
<td>0.2</td>
<td>0.2</td>
</tr>
<tr>
<td>v28</td>
<td>28.5</td>
<td>26.4</td>
<td>26.4</td>
</tr>
<tr>
<td>v29 N.A.</td>
<td>7.4</td>
<td>7.1</td>
<td></td>
</tr>
<tr>
<td>RMSE</td>
<td>11.7</td>
<td>10.1</td>
<td></td>
</tr>
</tbody>
</table>
Case 13 heat map

Case 14, 15

<table>
<thead>
<tr>
<th></th>
<th>13C-flux</th>
<th>ML</th>
<th>MFlux</th>
<th>13C-flux</th>
<th>ML</th>
<th>MFlux</th>
</tr>
</thead>
<tbody>
<tr>
<td>case 14</td>
<td>v1 EMP</td>
<td>100.0</td>
<td>99.5</td>
<td>100.0</td>
<td>100.0</td>
<td>99.0</td>
</tr>
<tr>
<td></td>
<td>v2 EMP</td>
<td>1.1</td>
<td>3.2</td>
<td>2.4</td>
<td>1.8</td>
<td>-0.2</td>
</tr>
<tr>
<td></td>
<td>v3 EMP</td>
<td>7.6</td>
<td>9.1</td>
<td>8.5</td>
<td>18.6</td>
<td>17.1</td>
</tr>
<tr>
<td></td>
<td>v4 EMP</td>
<td>7.6</td>
<td>7.8</td>
<td>8.5</td>
<td>18.6</td>
<td>18.4</td>
</tr>
<tr>
<td></td>
<td>v5 EMP</td>
<td>96.7</td>
<td>99.0</td>
<td>99.0</td>
<td>109.5</td>
<td>107.3</td>
</tr>
<tr>
<td></td>
<td>v6 EMP</td>
<td>84.5</td>
<td>86.7</td>
<td>86.7</td>
<td>90.7</td>
<td>88.5</td>
</tr>
<tr>
<td></td>
<td>v7 EMP</td>
<td>97.2</td>
<td>100.0</td>
<td>100.0</td>
<td>123.8</td>
<td>121.1</td>
</tr>
<tr>
<td></td>
<td>v8</td>
<td>106.7</td>
<td>104.5</td>
<td>104.5</td>
<td>96.9</td>
<td>99.2</td>
</tr>
<tr>
<td></td>
<td>v9</td>
<td>0.0</td>
<td>-1.0</td>
<td>-1.0</td>
<td>0.0</td>
<td>2.7</td>
</tr>
<tr>
<td></td>
<td>v10</td>
<td>97.7</td>
<td>97.7</td>
<td>97.6</td>
<td>97.1</td>
<td>97.1</td>
</tr>
<tr>
<td></td>
<td>v11</td>
<td>18.0</td>
<td>20.3</td>
<td>19.7</td>
<td>33.1</td>
<td>25.1</td>
</tr>
<tr>
<td></td>
<td>v12</td>
<td>9.1</td>
<td>10.4</td>
<td>10.4</td>
<td>21.6</td>
<td>20.3</td>
</tr>
<tr>
<td></td>
<td>v13</td>
<td>8.9</td>
<td>8.7</td>
<td>9.3</td>
<td>11.5</td>
<td>11.6</td>
</tr>
<tr>
<td></td>
<td>v14</td>
<td>5.6</td>
<td>6.1</td>
<td>5.9</td>
<td>11.8</td>
<td>11.3</td>
</tr>
<tr>
<td></td>
<td>v15</td>
<td>3.5</td>
<td>4.3</td>
<td>4.5</td>
<td>9.8</td>
<td>9.0</td>
</tr>
<tr>
<td></td>
<td>v16</td>
<td>5.6</td>
<td>5.6</td>
<td>5.9</td>
<td>11.8</td>
<td>12.3</td>
</tr>
<tr>
<td></td>
<td>v17 TCA cycle</td>
<td>86.4</td>
<td>86.4</td>
<td>81.5</td>
<td>78.6</td>
<td>78.6</td>
</tr>
<tr>
<td></td>
<td>v18 TCA cycle</td>
<td>86.4</td>
<td>84.5</td>
<td>81.5</td>
<td>78.6</td>
<td>76.8</td>
</tr>
<tr>
<td></td>
<td>v19 TCA cycle</td>
<td>86.0</td>
<td>67.1</td>
<td>75.0</td>
<td>79.0</td>
<td>72.7</td>
</tr>
<tr>
<td></td>
<td>v20 TCA cycle</td>
<td>73.1</td>
<td>71.1</td>
<td>66.0</td>
<td>65.0</td>
<td>67.1</td>
</tr>
<tr>
<td></td>
<td>v21 TCA cycle</td>
<td>70.0</td>
<td>67.0</td>
<td>72.6</td>
<td>59.0</td>
<td>62.1</td>
</tr>
<tr>
<td></td>
<td>v22 TCA cycle</td>
<td>70.0</td>
<td>67.0</td>
<td>72.6</td>
<td>59.0</td>
<td>62.0</td>
</tr>
<tr>
<td></td>
<td>v23 TCA cycle</td>
<td>70.0</td>
<td>66.8</td>
<td>67.4</td>
<td>59.0</td>
<td>62.4</td>
</tr>
<tr>
<td></td>
<td>v24 Glyoxylate</td>
<td>0.0</td>
<td>6.2</td>
<td>6.6</td>
<td>0.0</td>
<td>6.5</td>
</tr>
<tr>
<td></td>
<td>v25 ED</td>
<td>79.7</td>
<td>78.2</td>
<td>77.9</td>
<td>64.1</td>
<td>65.7</td>
</tr>
<tr>
<td></td>
<td>v26 ETOH</td>
<td>0.0</td>
<td>0.1</td>
<td>0.1</td>
<td>0.0</td>
<td>0.1</td>
</tr>
<tr>
<td></td>
<td>v27 LAC</td>
<td>0.0</td>
<td>-0.1</td>
<td>-0.1</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td></td>
<td>v28</td>
<td>23.7</td>
<td>21.5</td>
<td>21.5</td>
<td>27.3</td>
<td>25.2</td>
</tr>
<tr>
<td></td>
<td>v29</td>
<td>NA</td>
<td>11.7</td>
<td>11.7</td>
<td>NA</td>
<td>3.3</td>
</tr>
<tr>
<td></td>
<td>RMSE</td>
<td>4.1</td>
<td>3.4</td>
<td>2.9</td>
<td>3.6</td>
<td></td>
</tr>
</tbody>
</table>
Case 14 heat map
Case 15 heat map

<table>
<thead>
<tr>
<th>Case 16</th>
<th>13C-flux</th>
<th>ML</th>
<th>MFlux</th>
</tr>
</thead>
<tbody>
<tr>
<td>v1 EMP</td>
<td>100</td>
<td>99.4</td>
<td>100.0</td>
</tr>
<tr>
<td>v2 EMP</td>
<td>80</td>
<td>82.0</td>
<td>80.5</td>
</tr>
<tr>
<td>v3 EMP</td>
<td>91</td>
<td>92.6</td>
<td>91.5</td>
</tr>
<tr>
<td>v4 EMP</td>
<td>91</td>
<td>91.2</td>
<td>91.5</td>
</tr>
<tr>
<td>v5 EMP</td>
<td>187</td>
<td>184.7</td>
<td>184.7</td>
</tr>
<tr>
<td>v6 EMP</td>
<td>181</td>
<td>178.9</td>
<td>178.9</td>
</tr>
<tr>
<td>v7 EMP</td>
<td>192.5</td>
<td>195.3</td>
<td>195.3</td>
</tr>
<tr>
<td>v8</td>
<td>112</td>
<td>109.8</td>
<td>114.1</td>
</tr>
<tr>
<td>v9</td>
<td>46</td>
<td>48.9</td>
<td>42.1</td>
</tr>
<tr>
<td>v10</td>
<td>19.5</td>
<td>19.5</td>
<td>19.5</td>
</tr>
<tr>
<td>v11</td>
<td>19.5</td>
<td>20.3</td>
<td>19.4</td>
</tr>
<tr>
<td>v12</td>
<td>10.5</td>
<td>9.3</td>
<td>10.4</td>
</tr>
<tr>
<td>v13</td>
<td>9</td>
<td>8.9</td>
<td>9.0</td>
</tr>
<tr>
<td>v14</td>
<td>6</td>
<td>5.5</td>
<td>5.5</td>
</tr>
<tr>
<td>v15</td>
<td>4.5</td>
<td>5.3</td>
<td>4.9</td>
</tr>
<tr>
<td>v16</td>
<td>6</td>
<td>5.5</td>
<td>5.5</td>
</tr>
<tr>
<td>v17 TCA cycle</td>
<td>25</td>
<td>25.0</td>
<td>31.1</td>
</tr>
<tr>
<td>v18 TCA cycle</td>
<td>25</td>
<td>27.0</td>
<td>31.1</td>
</tr>
<tr>
<td>v19 TCA cycle</td>
<td>10.5</td>
<td>29.5</td>
<td>15.0</td>
</tr>
<tr>
<td>v20 TCA cycle</td>
<td>7</td>
<td>7.6</td>
<td>6.0</td>
</tr>
<tr>
<td>v21 TCA cycle</td>
<td>21.5</td>
<td>18.5</td>
<td>22.1</td>
</tr>
<tr>
<td>v22 TCA cycle</td>
<td>21.5</td>
<td>24.6</td>
<td>30.8</td>
</tr>
<tr>
<td>v23 TCA cycle</td>
<td>36</td>
<td>39.3</td>
<td>35.1</td>
</tr>
<tr>
<td>v24 Glyoxylate</td>
<td>14.5</td>
<td>18.0</td>
<td>16.1</td>
</tr>
<tr>
<td>v25 ED</td>
<td>0</td>
<td>1.6</td>
<td>0.1</td>
</tr>
<tr>
<td>v26 ETOH</td>
<td>28</td>
<td>28.0</td>
<td>24.8</td>
</tr>
<tr>
<td>v27 LAC</td>
<td>73</td>
<td>72.9</td>
<td>72.9</td>
</tr>
<tr>
<td>v28</td>
<td>-7.5</td>
<td>-5.3</td>
<td>-3.9</td>
</tr>
<tr>
<td>v29</td>
<td>NA</td>
<td>12.75</td>
<td>11.80</td>
</tr>
<tr>
<td>RMSE</td>
<td>4.0</td>
<td>3.0</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>case 17 X514</th>
<th></th>
<th>case 18 39E</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>13C-flux</td>
<td>ML</td>
<td>MFlux</td>
</tr>
<tr>
<td>v1 EMP</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
</tr>
<tr>
<td>v2 EMP</td>
<td>-1.0</td>
<td>3.0</td>
<td>-4.9</td>
</tr>
<tr>
<td>v3 EMP</td>
<td>65.0</td>
<td>66.4</td>
<td>65.0</td>
</tr>
<tr>
<td>v4 EMP</td>
<td>65.0</td>
<td>65.1</td>
<td>65.0</td>
</tr>
<tr>
<td>v5 EMP</td>
<td>163.0</td>
<td>160.8</td>
<td>160.9</td>
</tr>
<tr>
<td>v6 EMP</td>
<td>163.0</td>
<td>160.7</td>
<td>160.7</td>
</tr>
<tr>
<td>v7 EMP</td>
<td>171.0</td>
<td>169.5</td>
<td>169.5</td>
</tr>
<tr>
<td>v8</td>
<td>152.0</td>
<td>154.2</td>
<td>155.7</td>
</tr>
<tr>
<td>v9</td>
<td>21.0</td>
<td>23.8</td>
<td>21.4</td>
</tr>
<tr>
<td>v10</td>
<td>0.0</td>
<td>-0.4</td>
<td>4.9</td>
</tr>
<tr>
<td>v11</td>
<td>0.0</td>
<td>11.2</td>
<td>4.9</td>
</tr>
<tr>
<td>v12</td>
<td>-33.0</td>
<td>-10.3</td>
<td>-30.1</td>
</tr>
<tr>
<td>v13</td>
<td>33.0</td>
<td>32.9</td>
<td>35.0</td>
</tr>
<tr>
<td>v14</td>
<td>33.0</td>
<td>32.5</td>
<td>35.0</td>
</tr>
<tr>
<td>v15</td>
<td>33.0</td>
<td>32.2</td>
<td>35.0</td>
</tr>
<tr>
<td>v16</td>
<td>33.0</td>
<td>32.5</td>
<td>35.0</td>
</tr>
<tr>
<td>v17 TCA cycle</td>
<td>1.0</td>
<td>1.0</td>
<td>6.5</td>
</tr>
<tr>
<td>v18 TCA cycle</td>
<td>1.0</td>
<td>3.0</td>
<td>6.5</td>
</tr>
<tr>
<td>v19 TCA cycle</td>
<td>1.0</td>
<td>19.5</td>
<td>6.5</td>
</tr>
<tr>
<td>v20 TCA cycle</td>
<td>0.0</td>
<td>1.8</td>
<td>2.0</td>
</tr>
<tr>
<td>v21 TCA cycle</td>
<td>0.0</td>
<td>2.6</td>
<td>2.0</td>
</tr>
<tr>
<td>v22 TCA cycle</td>
<td>0.0</td>
<td>2.6</td>
<td>11.1</td>
</tr>
<tr>
<td>v23 TCA cycle</td>
<td>8.0</td>
<td>10.4</td>
<td>8.7</td>
</tr>
<tr>
<td>v24 Glyoxylate</td>
<td>0.0</td>
<td>-4.7</td>
<td>0.0</td>
</tr>
<tr>
<td>v25 ED</td>
<td>0.0</td>
<td>-1.4</td>
<td>0.0</td>
</tr>
<tr>
<td>v26 ETOH</td>
<td>129.0</td>
<td>128.9</td>
<td>127.8</td>
</tr>
<tr>
<td>v27 LAC</td>
<td>10.0</td>
<td>10.0</td>
<td>10.0</td>
</tr>
<tr>
<td>v28</td>
<td>-8.0</td>
<td>-6.0</td>
<td>-2.2</td>
</tr>
<tr>
<td>v29</td>
<td>-8.0</td>
<td>3.8</td>
<td>2.4</td>
</tr>
<tr>
<td>RMSE</td>
<td>6.5</td>
<td>4.1</td>
<td>6.6</td>
</tr>
</tbody>
</table>
Case 17 heat map
Case 18 heat map
Case 19


<table>
<thead>
<tr>
<th></th>
<th>13C-flux</th>
<th>ML</th>
<th>MFlux</th>
</tr>
</thead>
<tbody>
<tr>
<td>v1 EMP</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
</tr>
<tr>
<td>v2 EMP</td>
<td>-0.7</td>
<td>1.4</td>
<td>-2.8</td>
</tr>
<tr>
<td>v3 EMP</td>
<td>-2.2</td>
<td>-0.7</td>
<td>-2.0</td>
</tr>
<tr>
<td>v4 EMP</td>
<td>-2.2</td>
<td>-2.1</td>
<td>-2.0</td>
</tr>
<tr>
<td>v5 EMP</td>
<td>-2.5</td>
<td>-0.1</td>
<td>-3.7</td>
</tr>
<tr>
<td>v6 EMP</td>
<td>-3.4</td>
<td>-1.1</td>
<td>-3.7</td>
</tr>
<tr>
<td>v7 EMP</td>
<td>-3.8</td>
<td>-1.1</td>
<td>-1.1</td>
</tr>
<tr>
<td>v8</td>
<td>91.1</td>
<td>88.8</td>
<td>90.8</td>
</tr>
<tr>
<td>v9</td>
<td>84.0</td>
<td>81.1</td>
<td>78.0</td>
</tr>
<tr>
<td>v10</td>
<td>0.0</td>
<td>0.0</td>
<td>2.8</td>
</tr>
<tr>
<td>v11</td>
<td>0.0</td>
<td>11.4</td>
<td>2.8</td>
</tr>
<tr>
<td>v12</td>
<td>-0.4</td>
<td>-1.6</td>
<td>-0.3</td>
</tr>
<tr>
<td>v13</td>
<td>0.4</td>
<td>0.5</td>
<td>3.1</td>
</tr>
<tr>
<td>v14</td>
<td>-0.1</td>
<td>-0.6</td>
<td>0.4</td>
</tr>
<tr>
<td>v15</td>
<td>-0.3</td>
<td>-1.1</td>
<td>-0.7</td>
</tr>
<tr>
<td>v16</td>
<td>-0.1</td>
<td>-0.6</td>
<td>0.4</td>
</tr>
<tr>
<td>v17 TCA cycle</td>
<td>0.7</td>
<td>0.8</td>
<td>8.2</td>
</tr>
<tr>
<td>v18 TCA cycle</td>
<td>0.7</td>
<td>2.6</td>
<td>8.2</td>
</tr>
<tr>
<td>v19 TCA cycle</td>
<td>0.7</td>
<td>19.7</td>
<td>3.6</td>
</tr>
<tr>
<td>v20 TCA cycle</td>
<td>0.0</td>
<td>2.1</td>
<td>0.7</td>
</tr>
<tr>
<td>v21 TCA cycle</td>
<td>-0.8</td>
<td>2.2</td>
<td>5.2</td>
</tr>
<tr>
<td>v22 TCA cycle</td>
<td>-0.8</td>
<td>2.3</td>
<td>7.6</td>
</tr>
<tr>
<td>v23 TCA cycle</td>
<td>0.0</td>
<td>3.3</td>
<td>2.0</td>
</tr>
<tr>
<td>v24 Glyoxylate</td>
<td>0.0</td>
<td>6.5</td>
<td>4.6</td>
</tr>
<tr>
<td>v25 ED</td>
<td>0.0</td>
<td>0.9</td>
<td>0.0</td>
</tr>
<tr>
<td>v26 ETOH</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>v27 LAC</td>
<td>-100.0</td>
<td>-100.0</td>
<td>-99.9</td>
</tr>
<tr>
<td>v28</td>
<td>1.8</td>
<td>4.0</td>
<td>6.2</td>
</tr>
<tr>
<td>v29</td>
<td>-0.8</td>
<td>11.0</td>
<td>10.1</td>
</tr>
<tr>
<td>RMSE</td>
<td>5.1</td>
<td>4.0</td>
<td></td>
</tr>
</tbody>
</table>
Case 19 heat map
Case 20


<table>
<thead>
<tr>
<th>Case 20</th>
<th>13C-flux</th>
<th>ML</th>
<th>MFlux</th>
</tr>
</thead>
<tbody>
<tr>
<td>v1 EMP</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
</tr>
<tr>
<td>v2 EMP</td>
<td>-0.7</td>
<td>-2.6</td>
<td>-4.6</td>
</tr>
<tr>
<td>v3 EMP</td>
<td>-1.6</td>
<td>-3.1</td>
<td>-2.0</td>
</tr>
<tr>
<td>v4 EMP</td>
<td>-1.6</td>
<td>-1.8</td>
<td>-2.0</td>
</tr>
<tr>
<td>v5 EMP</td>
<td>-1.7</td>
<td>0.6</td>
<td>-2.7</td>
</tr>
<tr>
<td>v6 EMP</td>
<td>-2.3</td>
<td>-0.1</td>
<td>-2.7</td>
</tr>
<tr>
<td>v7 EMP</td>
<td>-1.9</td>
<td>1.0</td>
<td>1.0</td>
</tr>
<tr>
<td>v8</td>
<td>-7.9</td>
<td>-9.9</td>
<td>-7.8</td>
</tr>
<tr>
<td>v9</td>
<td>-100.0</td>
<td>-102.7</td>
<td>-100.0</td>
</tr>
<tr>
<td>v10</td>
<td>0.3</td>
<td>0.3</td>
<td>4.6</td>
</tr>
<tr>
<td>v11</td>
<td>0.3</td>
<td>11.7</td>
<td>4.6</td>
</tr>
<tr>
<td>v12</td>
<td>0.1</td>
<td>1.4</td>
<td>2.4</td>
</tr>
<tr>
<td>v13</td>
<td>0.2</td>
<td>0.2</td>
<td>2.2</td>
</tr>
<tr>
<td>v14</td>
<td>0.1</td>
<td>0.6</td>
<td>1.3</td>
</tr>
<tr>
<td>v15</td>
<td>0.0</td>
<td>0.8</td>
<td>1.1</td>
</tr>
<tr>
<td>v16</td>
<td>0.1</td>
<td>0.6</td>
<td>1.3</td>
</tr>
<tr>
<td>v17 TCA cycle</td>
<td>90.1</td>
<td>90.1</td>
<td>85.9</td>
</tr>
<tr>
<td>v18 TCA cycle</td>
<td>90.1</td>
<td>88.2</td>
<td>85.9</td>
</tr>
<tr>
<td>v19 TCA cycle</td>
<td>90.1</td>
<td>71.2</td>
<td>79.6</td>
</tr>
<tr>
<td>v20 TCA cycle</td>
<td>88.5</td>
<td>86.5</td>
<td>79.6</td>
</tr>
<tr>
<td>v21 TCA cycle</td>
<td>88.5</td>
<td>85.7</td>
<td>85.9</td>
</tr>
<tr>
<td>v22 TCA cycle</td>
<td>88.5</td>
<td>85.4</td>
<td>85.9</td>
</tr>
<tr>
<td>v23 TCA cycle</td>
<td>88.5</td>
<td>85.2</td>
<td>92.2</td>
</tr>
<tr>
<td>v24 Glyoxylate</td>
<td>0.0</td>
<td>6.5</td>
<td>6.3</td>
</tr>
<tr>
<td>v25 ED</td>
<td>0.0</td>
<td>1.4</td>
<td>0.0</td>
</tr>
<tr>
<td>v26 ETOH</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>v27 LAC</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>v28</td>
<td>4.2</td>
<td>6.4</td>
<td>6.4</td>
</tr>
<tr>
<td>v29</td>
<td>0.0</td>
<td>11.8</td>
<td>0.0</td>
</tr>
<tr>
<td>RMSE</td>
<td>5.1</td>
<td>3.6</td>
<td></td>
</tr>
</tbody>
</table>
Case 20 heat map
Attachment is two research papers on nanotechnology and electrospray published as first author separately.
Phytotoxicity of metal oxide nanoparticles is related to both dissolved metals ions and adsorption of particles on seed surfaces

Stephen G. Wu\textsuperscript{1,*}, Li Huang\textsuperscript{1,*}, Jennifer Head\textsuperscript{1}, Daren Chen\textsuperscript{1}, In Chul Kong\textsuperscript{#,2}, and Yinjie J. Tang\textsuperscript{#,1}

1. Department of Energy, Environmental and Chemical Engineering, Washington University, St. Louis, Missouri 63130, USA
2. Department of Environmental Engineering, Yeungnam University, Kyungsan City, Kyungbuk 712-749, Republic of Korea

Running title: Investigation of metal oxide NP Phytotoxicity

*: Wu and Huang contributed equally to this study.

Corresponding author

ICK: Email: ickong@ynu.ac.kr, Tel: 82-53-810-2546; Fax: 82-53-810-4624.

YJT: Email: yinjie.tang@seas.wustl.edu, Tel: 1-314-935-3441, Fax: 1-314-935-5464.
Abstract

This study assesses the biological effects of nanoparticles (NPs) based on seed germination and root elongation tests. Lettuce, radish and cucumber seeds were incubated with various metal oxide NPs (CuO, NiO, TiO$_2$, Fe$_2$O$_3$, Co$_3$O$_4$), of which only CuO and NiO showed deleterious impacts on the activities of all three seeds. The measured EC$_{50}$ for seed germinations were: lettuce seed (NiO: 28 mg/L; CuO: 13 mg/L), radish seed (NiO: 401 mg/L; CuO: 398 mg/L), and cucumber seed (NiO: 175 mg/L; CuO: 228 mg/L). Phytotoxicity of TiO$_2$, Fe$_2$O$_3$ and Co$_3$O$_4$ to the tested seeds was not significant, while Co$_3$O$_4$ NP solution (5 g/L) was shown to improve root elongation of radish seedling. Metal oxide NPs tended to adsorb on seed surfaces in the aqueous medium and released metal ions near the seeds. Therefore, metal oxide NPs had higher phytotoxicity than free metal ions of the equivalent concentrations. Further, the surface area-to-volume ratio of seeds may also affect NPs phytotoxicity, whereby small seeds (i.e., lettuce) were the most sensitive to toxic CuO and NiO NPs in our experiments.

Key words: CuO, EC$_{50}$, NiO, root elongation, metal ions, seeds germination
Introduction

As applications for metal oxide nanoparticles (NPs) are employed by industry, the release of nanomaterials into the environment may pose severe threats for ecological systems and human health [(Roco 2005; Lin and Xing 2008; Klaine et al. 2008; Marambio-Jones and Hoek 2010)]. Risk assessments of nano-toxicities have already attracted public attention [(Roco 2005)]. Toxic effects of NPs on microorganisms and animals have also been reported [(Marambio-Jones and Hoek 2010; Wang et al. 2010b; Ji et al. 2011; Ma et al. 2010a; Wang et al. 2006b; Navarro et al. 2008a; Menard et al. 2011)], where metal oxide nanoparticles are the most extensively studied. Their toxicities are attributed to three mechanisms: 1. Generation of reactive oxygen species (ROS), which can damage the cell membrane; 2. Penetration of nanoparticles into the cell where they interfere with intracellular metabolism (nano-Trojan horse effect) [(Limbach et al. 2007)]; 3. Release of metal ions that hinder enzyme functions. Moreover, the phytotoxicity profile of NPs has also been investigated by researchers via seed germination and root elongation tests which evaluate the acute effects of NPs on plant physiologies [(Di Salvatore et al. 2008)]. For instance, alumina and zinc oxide NPs have been applied to different plant species [(Lin and Xing 2007; Yang and Watts 2005)]. Inhibition of seed germination and root elongation has been found to be highly dependent on both plant type and NP properties. This paper explores the impacts of additional metal oxide NPs on seed activities. In particular, we investigate three common vegetable seeds after they were incubated in aqueous NP-containing solutions: lettuce (Lactuca sativa) seed (length/width: 3 mm /1 mm); radish (Raphanus sativus) seed (length/width: 3 mm /3 mm) and cucumber (Cucumis sativus) seed (length/width: 8 mm /6 mm). This work aims to increase understanding of both NPs phytotoxicity on various edible plants and the potential impact of NPs on agricultural processes [(Mondal et al. 2011; Rico et al. 2011)].
Materials and Methods

Chemicals. All chemicals used were reagent grade and purchased from Sigma (St. Louis, MO, US) or Fisher (Pittsburg, PA, US). TiO$_2$ NP (30-50 nm), Fe$_2$O$_3$ NPs (20-40 nm), CuO NPs (30-50 nm), NiO NPs (30 nm) and Co$_3$O$_4$ NPs (10-30 nm) were obtained from Nanostructured & Amorphous Materials, Inc. (Houston, TX, US). The pH of germination solutions (containing deionized water and NP suspensions) was adjusted to 7 for all toxicity studies done in aqueous phases.

Seed Germination and Root Elongation Assay. Lettuce, radish and cucumber seeds purchased from Ferry-Morse Seed Co. (Fulton, KY, US) were used in this study (Lettuce, Black Seeded Simpson, 2846; Radish, Icicle, Short Top, 3236; Cucumber, Marketmore 76, 2646). All three species are commonly used and recommended for phytotoxicity tests [(Rivetta et al. 1997; Wang et al. 2001; U.S.EPA 1996)]. Seeds were first sterilized by soaking them in 3% H$_2$O$_2$ solution for 1 min and then rinsing twice with deionized water (dH$_2$O). After, seeds were placed into dH$_2$O (control) or certain NP solutions and shaken gently for two-hours [(Lin and Xing 2007)]. All seeds were subsequently transferred into 15 mm × 100 mm Petri dishes containing one piece of filter paper (90 mm in diameter, Whatman NO.1). 10 seeds of radish and cucumber or 15 seeds of lettuce were evenly spaced on top of the filter paper in each Petri dish. The dishes were filled with 5 ml of dH$_2$O or NP solutions and sealed before being incubated at 25 °C in dark conditions [(Reddy and Singh 1992; El-Temsah and Joner 2010)]. After 3 days of incubation, the root length of each seed was measured. Experimental procedures are summarized in Figure 1. In this study, root length greater than 1 cm for lettuce seeds and 2 cm for radish and cucumber seeds was considered positive for germination based on our preliminary experiments. For each
condition, experiments were conducted in triplicate, from which standard deviations were calculated.

**Data Analysis.** Three parameters were adopted in this analysis to evaluate the conditions of seed germination: Relative germination rate, Germination Index and EC$_{50}$ value. They were calculated based on the following equations according to previous reports [(Barrena et al. 2009; Thompson et al. 2001)]:

- **Relative germination rate** = \( \frac{\text{Seeds germinated in test sample}}{\text{Seeds germinated in control}} \times 100 \)

- **Relative root elongation** = \( \frac{\text{Mean root length in test sample}}{\text{Mean root length in control}} \times 100 \)

- **Germination Index** = \( \frac{\text{Relative germination rate} \times \text{Relative root elongation}}{100} \)
EC$_{50}$ is defined as the effective concentration of a certain drug/chemical that reaches half of its maximal effects or reduces growth of the control by 50%. We employed the software provided by the USEPA ([19] http://www.epa.gov/eerd/stat2.htm/tsk) which utilizes the Trimmed Spearman-Karber Method to calculate EC$_{50}$ values for different chemicals [(Hamilton et al. 1977)]. Student’s t-test was performed to analyze the variations in root length and germination rate between different treatments and control groups. Statistics Toolbox of Matlab (MathWorks, MA, US) was employed to conduct all statistical analyses and statistically significant was defined at the level of P < 0.05.

**Determination of metal ions released from NP suspensions.** To measure the concentration of metal ions released from NP solutions, aliquots of all five NP suspensions were drawn after the suspensions were incubated at room temperature for 2 hours. The extracts were centrifuged at 19,000 g for 20 min, and supernatants were collected and filtered with 0.22 μm nylon filters (GE Water & Process Technologies, CT, US). Inductively coupled plasma mass spectroscopy (ICP-MS, Agilent, CA, US) was used to conduct concentration assays of metal ions, and duplicated samples were measured for each condition.

**Protocols for Scanning Electron Microscope (SEM) and Dynamic Lighting Scattering (DLS).** Seeds sprayed with NPs or incubated with NP suspensions were dried overnight in a fume hood. They were then coated with gold nanoparticles by a low vacuum sputter coater (SPI supplies, PA, US) prior to image taking. Images of seed surfaces were taken with a scanning electron microscope (SEM) (Nova 2300 FEI, OR, US) and Zeta potential of NP suspensions was determined by dynamic lighting scattering (Malvern Instruments, Worcestershire, UK) after 30 minutes of incubation in room temperature.

**Results and Discussion**
The toxicities of different metal oxide NPs at various concentrations on lettuce, radish, and cucumber seeds were tested. Seeds incubated in dH$_2$O (pH = 7) were considered as the control upon which all statistical analysis was performed. From results shown in Table 1 and Figure 2, CuO and NiO NPs were far more toxic than the other three NPs on all three species of seeds, while lettuce seeds were the most sensitive to NPs in terms of germination. Our results showed that the toxicities of the NPs were also dependent upon the plant species, which was in accordance with a previous report [(Lin and Xing 2007)]. The relative toxicities based on the germination index (combined seed germination and root elongation) for the tested NPs were listed below:

**Lettuce**  
CuO > NiO >> Fe$_2$O$_3$ > TiO$_2$ ≈ Co$_3$O$_4$

**Radish**  
NiO > CuO >> TiO$_2$ > Fe$_2$O$_3$ > Co$_3$O$_4$

**Cucumber**  
NiO > CuO >> Fe$_2$O$_3$ > TiO$_2$ > Co$_3$O$_4$

Interestingly, Co$_3$O$_4$ NP solution did not inhibit the germination of cucumber seeds and even improve root elongation of radish seedling at high concentrations (5 g/L). Previous studies have provided similar reports of the positive effects of NPs on germination and growth of plants. For example, TiO$_2$ and SiO$_2$ NPs are found to enhance both the germination and growth of *Glycine max* seeds [(Lu et al. 2002)], carbon nanotubes (CNT) are discovered to improve germination and root elongation of tomato seeds (Khodakovskaya et al. 2009)], and Nano-Al are shown to augment root elongation of radish and rape seedling (Lin and Xing 2007)]. Such observations are likely due to an increased water uptake by seeds in the presence of high concentrations of NPs (Nair et al. 2010)].
The biological effects of NPs in aqueous solutions are closely associated to the concentration of released metal ions [(Ji et al. 2011; Navarro et al. 2008b)]. In this study, we measured the concentrations of metal ions released from all five types of NPs. We did not detect any metal ions released from TiO$_2$ NP solution, while Fe$_2$O$_3$ and Co$_3$O$_4$ NPs both released trace metal ions. For example, the aqueous solution with Co$_3$O$_4$ NPs contained ~2 mg/L cobalt ion, but its inhibition of seed activity was minimal. Similarly, both Cu and Ni ions were released from the metal oxide NPs during incubation with the seeds (Table 2). To compare phytotoxicity between metal ions and NPs, we assessed seed activity in copper chloride and nickel chloride solutions and determined their EC$_{50}$ values. When CuCl$_2$ or NiCl$_2$ solutions were used to treat seeds (Table 2), the EC$_{50}$ concentrations of Cu$^{2+}$ and Ni$^{2+}$ were 5 ~ 8 mg/L and 9 ~ 19 mg/L, respectively. However, at their EC$_{50}$ concentrations, CuO or NiO NPs released much lower free metal ions (less than 2 mg/L). For example, a 13 mg/L CuO NP solution was able to strongly inhibit lettuce seed germination, while the released Cu$^{2+}$ concentration in the culture medium was only ~ 0.2 mg/L. Therefore, the phytotoxicity of metal oxide NPs is not only due to their dissolved metals ions, but also to their interactions with the seed/root surface.

It has been widely accepted that smaller NPs would have higher surface energy and thus cause more toxic to the cell [(Krug and Wick 2011)]. However, metal oxide NPs often agglomerate in the aqueous phase to minimize surface energy, and disaggregating is extremely difficult [(Lin and Xing 2007; Yang and Watts 2005)]. The actual size of our tested NPs in the aqueous solution was therefore up to 1 micrometer due to agglomeration (Table 3 and Figure 3). Previous studies reported that increasing the size of particle aggregates would reduce the toxic effect of the metal oxide particles [(Lin and Xing 2007; Yang and Watts 2005)]. On the other hand, suspended metal oxide NPs tend to agglomerate and accumulate on root/seed surfaces.
[(Nair et al. 2010)], and phytotoxicity in our tests was not likely caused by mono-dispersed NPs. Instead, we observed that a large amount of NPs (e.g., TiO$_2$ or CuO) adsorbed on the surface of the seeds in all experiments (Figure 3). The main factors contributing to such adsorption can be concluded as increased surface area due to a rough seed surface, surface charges of NP agglomeration (e.g., 1000 mg/L of CuO NPs: $-23.5 \pm 94.5$ mV, determined by DLS) and hydrophobic interactions between the NPs and the seed coat. Variations in the ratio of lipid to fatty acid content and the wax to fatty acid layer of the seed coat would affect the strength of such hydrophobic interactions, and thus NPs phytotoxicity [(Zhu et al. 2005; Zeng et al. 2005; Hu et al. 1994)]. The adsorption of NPs on the seed surface can enhance the effect of locally concentrated ions (released from NPs) on seed activities. The adsorption of metal oxide NPs on the seeds’ surface also explains why small-size lettuce seeds are particularly sensitive to NP phytotoxicity. Because of the relatively high ratio of surface area to volume, more NPs per unit volume can be absorbed on the seed surface, thus increasing their toxic effect (Krug and Wick 2011; Stark 2011). Therefore, the toxic NPs are more inhibitory on the germination of lettuce seeds (Figure 4).

**Conclusion**

Our experiments determined the impact of five different nanoparticles on common plant seeds. It was discovered that smaller sized seeds, such as lettuce seeds, are more sensitive to toxic NPs. Additionally, this study shows that engineered metal oxide nanoparticles may hold significant potential applications in agriculture and gardening, as they may selectively inhibit unwanted plants (such as weeds), kill harmful fungi and bacteria in plant fields, and release essential metal elements for plant growth.
Acknowledgments

We appreciate valuable suggestions given by Dr. Hui Wei from the University of Illinois at Urbana-Champaign. This research was supported by the National Research Foundation of Korea (2011-0026754) through the Ministry of Education, Science and Technology given to I. C. Kong. This research was also supported by Washington University MAGEEP program.

References


73. El-Temsah, Y. S.; Joner, E. J., Impact of Fe and Ag nanoparticles on seed germination and differences in bioavailability during exposure in aqueous suspension and soil. Environ Toxicol 2010, n/a-n/a.


Figure Legend

Figure 1. Flow chart of experimental procedures

Figure 2. Effects of NPs on seed germination and elongation; Red line: relative germination rate; Blue dashed line: germination index.

Figure 3. SEM images for NPs/lettuce seeds. In the aqueous phase, the SEM image shows that metal oxide NPs (TiO$_2$ NPs 1000 mg/L) (a) and (CuO NPs 1000 mg/L) were adsorbed on the seed surface (b).

Figure 4. Effects of CuO NPs on seed germination and root elongation (incubation at 25 °C in dark for 3 days, NPs could be observed on the seed surface.)

Figure 4-1. Lettuce seeds (a) Incubated in dH$_2$O; (b) Incubated in 500 mg/L of CuO NPs.

Figure 4-2. Radish seeds (a) Incubated in dH$_2$O; (b) Incubated in 500 mg/L of CuO NPs.

Figure 4-3. Cucumber seeds (a) Incubated in dH$_2$O; (b) Incubated in 500 mg/L of CuO NPs.
Figure 1. Flow chart of experimental procedures

- Prepare and sonicate NP solutions
- Rinse seeds with 3% H₂O₂, then wash with dH₂O
- Put seeds into NP solutions, shake gently for two hours
- Transfer seeds into petri dishes and add 5 ml of NP solution
- Incubate @ 25°C in dark for 3 days
- Harvest seeds and measure the length of their roots
Figure 2. Effects of NPs on seed germination and elongation. Red line: relative germination rate; Blue dash line: germination index.
Figure 3. SEM images for NPs/lettuce seeds. In the aqueous phase, the SEM image shows that metal oxide NPs (TiO$_2$ NPs 1000 mg/L) (a) and (CuO NPs 1000 mg/L) were adsorbed on the seed surface (b).
Figure 4. Effects of CuO NPs on seed germination and root elongation (incubation at 25 °C in dark for 3 days, NPs could be observed on the seed surface.)

1. Lettuce seeds (a) Incubated in dH₂O; (b) Incubated in 500 mg/L of CuO NPs.

2. Radish seeds (a) Incubated in dH₂O; (b) Incubated in 500 mg/L of CuO NPs.

3. Cucumber seeds (a) Incubated in dH₂O; (b) Incubated in 500 mg/L of CuO NPs.
Table 1. Effects of NPs on seeds activities

<table>
<thead>
<tr>
<th>NP</th>
<th>Lettuce</th>
<th>Radish</th>
<th>Cucumber</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>EC&lt;sub&gt;50&lt;/sub&gt; (mg/L)</td>
<td>GI affected by 1000 mg/L NP</td>
<td>EC&lt;sub&gt;50&lt;/sub&gt; (mg/L)</td>
</tr>
<tr>
<td>CuO</td>
<td>12.9</td>
<td>-100%*</td>
<td>397.6</td>
</tr>
<tr>
<td>NiO</td>
<td>27.9</td>
<td>-100%*</td>
<td>400.7</td>
</tr>
<tr>
<td>Fe&lt;sub&gt;2&lt;/sub&gt;O&lt;sub&gt;3&lt;/sub&gt;</td>
<td>&gt; 5000</td>
<td>-55.0%*</td>
<td>&gt; 5000</td>
</tr>
<tr>
<td>TiO&lt;sub&gt;2&lt;/sub&gt;</td>
<td>&gt; 5000</td>
<td>-36.2%</td>
<td>&gt; 5000</td>
</tr>
<tr>
<td>Co&lt;sub&gt;3&lt;/sub&gt;O&lt;sub&gt;4&lt;/sub&gt;</td>
<td>&gt; 5000</td>
<td>-43.6%</td>
<td>&gt; 5000</td>
</tr>
</tbody>
</table>

GI – Germination Index; ‘+’ - enhancement, ‘-’ – inhibition, ‘*’ – significant difference

Table 2. EC<sub>50</sub> values of Cu<sup>2+</sup>/Ni<sup>2+</sup> vs. released ions from NPs at their EC<sub>50</sub> concentration

<table>
<thead>
<tr>
<th>Seeds</th>
<th>Lettuce</th>
<th>Radish</th>
<th>Cucumber</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>EC&lt;sub&gt;50&lt;/sub&gt; for ions (mg/L) *</td>
<td>Cu&lt;sup&gt;2+&lt;/sup&gt;</td>
<td>Ni&lt;sup&gt;2+&lt;/sup&gt;</td>
</tr>
<tr>
<td>EC&lt;sub&gt;50&lt;/sub&gt; for ions (mg/L) *</td>
<td>Cu&lt;sup&gt;2+&lt;/sup&gt;</td>
<td>4.9 [3.9, 6.0]</td>
<td>8.0 [5.8, 11.0]</td>
</tr>
<tr>
<td>EC&lt;sub&gt;50&lt;/sub&gt; for ions (mg/L) *</td>
<td>Ni&lt;sup&gt;2+&lt;/sup&gt;</td>
<td>8.8 [6.5, 11.9]</td>
<td>18.7 [15.9, 22.0]</td>
</tr>
<tr>
<td>Released ions in solution from CuO and NiO NPs (mg/L) **</td>
<td>Cu&lt;sup&gt;2+&lt;/sup&gt;</td>
<td>0.20 ± 0.16 (13)</td>
<td>1.75 ± 0.45 (400)</td>
</tr>
<tr>
<td>Released ions in solution from CuO and NiO NPs (mg/L) **</td>
<td>Ni&lt;sup&gt;2+&lt;/sup&gt;</td>
<td>0.26 ± 0.19 (28)</td>
<td>1.97 ± 0.64 (400)</td>
</tr>
</tbody>
</table>

* Values of 95% confidence interval of free metal ions were in the bracket [].
** Concentrations of released metal ions from NP solutions incubated with different seeds: NPs in the experiments were at the concentrations of their approximate respective EC<sub>50</sub> values (in parentheses). Data were averaged based on duplicated samples.

Table 3. Size distribution of typical metal oxide NP solutions

<table>
<thead>
<tr>
<th>Total *NP in solution (mg/L)</th>
<th>CuO</th>
<th>NiO</th>
<th>Fe&lt;sub&gt;2&lt;/sub&gt;O&lt;sub&gt;3&lt;/sub&gt;</th>
<th>Co&lt;sub&gt;3&lt;/sub&gt;O&lt;sub&gt;4&lt;/sub&gt;</th>
<th>TiO&lt;sub&gt;2&lt;/sub&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>100</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td><strong>Average size (nm)</strong></td>
<td>984</td>
<td>576</td>
<td>246</td>
<td>440</td>
<td>562</td>
</tr>
</tbody>
</table>

* NPs suspension were prepared in dH<sub>2</sub>O (pH = 7)
** Average sizes (Z average) were determined by DLS after NPs incubated at room temperature for 30 min.
Electrospray Facilitates the Germination of Plant Seeds

Stephen G. Wu*, Li Huang*, Jennifer Head, Madelyn Ball, Yinjie J. Tang, Da-Ren Chen#

Department of Energy, Environmental and Chemical Engineering, Washington University in St. Louis, St. Louis, Missouri 63130, USA

Abstract

We proposed a new approach to enhance the plant seed germination via the electrospray of nanoparticles (NPs). A single-capillary electrospray system with a particle deposition stage (where seeds are placed) was set up for this investigation. For demonstration, lettuce (*Lactuca sativa*) seeds were bombarded by TiO₂ NPs via the electrospray for 2~4 minutes in order to promote their germination. Based on our study, the enhancement on germination was significant in cases with aged seeds or seeds placed in an unfavorable growth condition (e.g., low pH medium). The electrospray of other NPs (i.e., Au and CuO) were also shown to be effective in enhancing the germination of aged lettuce seeds. TEM (Transmission Electron Microscopy) and SEM-EDX (Scanning Electron Microscopes and Energy-Dispersive X-ray Spectroscopy)
analyses suggested that sprayed NPs penetrate the seed coat via the frequent bombardment of NPs at high speeds, thus breaking the coat-imposed seed dormancy. The enhancement on the germination of grass seeds was also observed in this study. The proposed seed treatment may have the potential to improve the germination of various recalcitrant crop seeds.

**Keywords**: aerosolized TiO$_2$ NP, crop seeds, lettuce, shelf life, TEM, seed dormancy
1. Introduction

In recent years, Nanotechnology and Nanoparticle-related research have undergone rapid growth in various fields, including nanomedicines, drug delivery, biomedical imaging and sensing, and solar energy conversion (Yoo et al. 2011; Guo and Dong 2011; Wei et al. 2007). Nanoparticles are defined as objects with at least two dimensions less than 100 nm. The unique surface area and solubility of nanoparticles distinguish them from their molecular and bulk counterparts, contributing to various biological effects (Menard et al. 2011; Nel et al. 2006). In work related to plants, several studies have focused on the translocation of NPs in plants (Lin and Xing 2008; Khodakovskaya et al. 2009; Ma et al. 2010b). For instance, nanomaterials can be used as a vector to deliver DNAs or other chemicals into plant cells and tissues (Nair et al. 2010; Torney et al. 2007; González-Melendi et al. 2008; Liu et al. 2009; Martin-Ortigosa et al. 2012). There are also extensive studies on nanotoxicity on plants in the hopes of advancing nanobiotechnology into agricultural applications while relieving the public concern of its potential risk (Rico et al. 2011). Zheng et al. (2005) has showed that a low dosage of TiO$_2$ NPs had no harmful effects on spinach plants, but rather promoted photosynthesis and nitrogen metabolism that benefited the plant’s growth (Zheng et al. 2005). Multi-walled carbon nanotubes and zinc oxide NPs were found to be able to stimulate the seed germination, thus enhancing the plant’s growth in the aqueous culture (Khodakovskaya et al. 2009; Prasad et al. 2012). More, a series of approaches (i.e., genetic, photo-thermal and photo-acoustic methods) were combined to characterize the interactions between multiwalled carbon nanotubes and tomato tissues, providing new insights into the gene transcription regulations of plants under the influence of nanomaterials (Khodakovskaya et al. 2011). However, colloidal suspensions of NPs were used in the previous literature. It is known that NPs in suspended solutions tend to agglomerate in general, resulting
in the reduction of their nanoscale effect (Nel et al. 2006). It is thus more desirable to have the NPs in their singlet form to study the effect of NPs. Electrospray has been demonstrated to accomplish the above task (Kim et al. 2010).

In this study, we employed a single-capillary electrospray setup to simultaneously disperse and deliver individual NPs onto the surface of plant seeds. TiO$_2$ NPs were applied in this investigation because of their low cost and low toxicity. _Lactuca sativa_ (an edible lettuce) was selected as an example plant to demonstrate the feasibility of the proposed treatment for crop seeds. Tests on other NPs (Au and CuO) and grass seeds were also performed to support our idea.

## 2. Experiment

### 2.1 Electrospray setup

The detail of the experimental setup has been described in the work of Wu et al. (2010). A brief description is provided herein for the reference. The schematic diagram of the experimental setup is shown in Figure 1. The single-capillary electrospray setup consisted of four components: a spray head, a particle deposition stage, a high voltage power supply (Bertan Model 230), and an optical monitor system. The electrospray head was a single capillary, connected to a syringe driven by a programmable Harvard syringe pump (PHD 2000). Sprayed TiO$_2$ suspension was fed through the spray head at 2 μL/min. A non-uniform electrical field was established between the spray head and deposition stage by applying a positive high voltage on the spray nozzle and electrically grounding the stage. The deposition stage provided a platform on which the lettuce seeds (typically 30 lettuce seeds for each run) were exposed to the NP electrospray. The distance between the capillary tip and the seed platform was kept at 2.0 cm. In
this study, a typical working voltage was about 7 kV ~ 10 kV for operating the electrospray at the so-called cone-jet mode (Chen et al. 1995). The optical monitor system, which included a microscopic lens (InfiniGage, Infinity Photo-Optical Co. Japan), CCD camera (XC-ST 70, Infinity Photo-Optical Co. Japan) and a LCD screen, magnified the liquid meniscus at the capillary exit in order to monitor the cone-jet operation.

All spray suspensions were freshly prepared by dispersing the NPs (via 1 min of sonication, 15 W, Misonix XL-2000 Ultrasonic Liquid Processors, NY, US) in 1 mM of a sodium acetate buffer (pH = 7) to ensure that the conductivity of sprayed suspensions was in the range of 150-200 μS/cm.

2.2 Chemicals and Materials.

TiO$_2$ NPs (30-50 nm, rutile) and CuO NPs (30-50 nm) were obtained from Nanostructured & Amorphous Materials, Inc. (Houston, TX, US), while Au NPs (50 nm) was obtained from BBInternational Inc. (Cardiff, UK). The lettuce seeds (*Lactuca sativa*, Black Seeded Simpson, #2846) were purchased from Ferry-Morse Seed Co. (Fulton, KY, US), while the yarrow (*Achillea millefolium*) and common ragweed (*Ambrosia artemisiifolia*) seeds were purchased from Herbiseed Company (Twyford, UK).

To investigate the effect of the electrospray treatment on seed germination in an acidic or basic environment (e.g., to mimic the growth conditions in acidic & alkaline soil), we prepared the following buffer solutions for the seed culture: 2 mM of citric acid buffer for a pH of 3; 2 mM of MES (2-(N-morpholino) ethanesulfonic acid) buffer for a pH of 5; 2 mM of HEPES (4-2-hydroxyethyl-1-piperazineethanesulfonic acid) buffer for a pH of 7; 2 mM of Tricine (N-tris
(hydroxymethyl) methylglycine) buffer for a pH of 9; and 2 mM of N-cyclohexyl-3-aminopropanesulfonic acid buffer for a pH of 11 (Reddy and Singh 1992).

2.3 Determination of size distribution of Electrosprayed NPs.

The size distribution of TiO$_2$ NPs after electrospray was characterized by a scanning mobility particle sizer (SMPS, TSI, USA) via the electrospray setup described by Chen et al (1995). The size distribution of the TiO$_2$ NPs in suspension was determined by dynamic light scattering (DLS) (Zetasizer, Malvern Instruments, Worcestershire, UK).

2.4 Protocol for SEM and TEM Images of Treated Seeds

To observe the NPs’ bombardment on the seeds, the sprayed lettuce seeds were air dried and subjected to scanning electron microscopy (SEM, Nova 2300 FEI, OR, US). When necessary, samples were coated with gold by a low vacuum sputter coater (SPI supplies, PA, US) to increase the surface conductivity before imaging. To validate the penetration of nanoparticles through the seed coat, a TEM (Transmission Electron Microscope, JEOL 1200 EX, MA, US) was used to image the cross sections of treated seeds. Seeds were first fixed overnight at 4 °C in a phosphate buffer solution containing 2% paraformaldehyde and 2.5% glutaraldehyde (pH 7.2), followed by post-fixing with 1% osmium tetroxide for 2 hours after a phosphate buffer wash. Subsequently, these samples were stained with 1% aqueous uranyl acetate overnight at 4 °C. After dehydration with sequential ethanol concentrations ranging from 50 to 100%, sections of each sample were cut with a Leica Ultracut UCT ultramicrotome (Leica Microsystems Inc., Bannockburn, IL, US) and placed on grids for TEM imaging. To further confirm the existence of nanoparticles in the seed sections, the sections were characterized by an energy dispersive X-ray spectroscopy, coupled with SEM (SEM-EDX) for elemental analysis of titanium.
2.5 Seed germination.

Thirty lettuce seeds were placed on the electrically-grounded deposition stage to undergo NP electrospray treatment for time periods of approximately 5 minutes. After the spray, all seeds were transferred into petri dishes (15 mm x 100 mm) containing 5 mL of medium solution. Fifteen lettuce seeds were incubated in each petri dish to ensure adequate space to germinate and grow. All dishes were sealed and incubated at 25 °C in the dark (Reddy and Singh 1992). After the end of the germination period (usually three days), the seedling’s length were measured (note: majority of lettuce seeds were germinated in the first three days of incubation). The lettuce seeds with seedling lengths longer than 1 cm were considered to be well-germinated seeds. A total of 60 seeds were used for each treatment condition for analysis. Similar protocols were performed for the other seeds with slight variances in seed number per petri dish (i.e., Yarrow seed: about 120 per petri dish/2 weeks; Common ragweed seed: 50 per petri dish/2 weeks.).

The impact of NP electrospray was evaluated based on the seed germination percentage, calculated by the following equation:

\[
\text{Germination percentage} = \frac{\text{Number of seeds well germinated}}{\text{Total number of seeds}}
\]

Note that we used the germination percentage as the sole parameter to evaluate the effect of the NP electrospray treatment. The Statistics Toolbox of MATLAB was employed to conduct the data analyses based on the Z test for germination percentage, where statistically significant was defined as \( P < 0.05 \).

3. Results and Discussion

294
3.1 The electrospray of TiO$_2$ NP onto plant seeds

Figure 2 shows the size distribution measurement of TiO$_2$ NPs in aqueous solutions. It is evidenced that TiO$_2$ NPs in aqueous solution formed agglomerates up to several µm in size. The agglomeration of NPs reduces the particle size effect, particularly on cellular function (Wu et al. 2010). By adjusting the electrical conductivity and the feeding flow rate of TiO$_2$ NP suspensions, the single-capillary electrospray operated at the cone-jet mode ensured the production of monodisperse droplets in sub-micrometer and nanometer size ranges (Chen et al. 1995; Jaworek 2007). When the concentration of TiO$_2$ NPs in the solutions was diluted, the solvent in the droplets evaporated during electrospray so that singlet TiO$_2$ NPs could be dispersed onto seed surface.

Figure 2 shows the size distributions measured by the SMPS when freshly prepared TiO$_2$ NP suspensions of 1 g/L are electrosprayed. The measured particle size distribution exhibited a narrow peak at 36.0 nm, suggesting that most particles were singly dispersed after the electrospray process. Once electrosprayed, gas-borne NPs were accelerated by the presence of a DC electric field. The terminal velocity of sprayed NPs was estimated to be in the range of 100 to 500 m/s prior to bombarding the seed coat (Pui and Chen 2000). The penetration of NPs through the seed coat under the proposed treatment was verified by the SEM images (shown in Figure 3). To our knowledge, this is the first report demonstrating the application of using aerosolized NPs to penetrate plant seed coats.

The seed coat, consisting of layers of the testa and endosperm envelope (Welbaum et al. 1998), provides protection against the entry of parasites and mechanical injury. However, the coat may impose the seed dormancy (Zeng et al. 2005). To improve seed germination, we
employed NPs with high electrical charges to facilitate the break of seed coat by accelerating them in a DC electrical field. Different from the delivery of particles into plant cells via high-pressure gas (Torney et al. 2007; Gordon-Kamm et al. 1990), the electrospray accelerates particles primarily through the space charge effect (due to the presence of highly charged particles in high concentration) (Chen et al. 2000). The NP’s entry into seeds via electrospray demonstrates the potential of this proposed method for delivering various materials (DNAs or plant hormones) into embryos (Gu et al. 2011). By tuning the electrical field strength and controlling the charges on the NPs, a broad range of particle velocities can be achieved for the bombardment of targeted organelles. The optimal speeds needed for the successful delivery of nanomaterials (varying in size, density, and shape) into various types of seeds would be an interesting topic to explore in the near future.

3.2 Seed germination enhancement

Two factors may affect the germination of seeds. First factor, if the seeds were stored for a long period of time, their germination are typically reduced (i.e., become more recalcitrant). Second factor is the soil pH, an important factor influencing seed germination, and plants have to adapt in acidic or alkaline environments. Thereby, we tested the effectiveness of NP treatment for seed germination under unfavorable conditions. Figure 4 summarizes the experimental results of seed germination of lettuce seeds when treated with TiO$_2$ NPs electrospray and incubated in buffer solutions. Figure 4a is the germination percentage of aged lettuce (stored for 10 months) treated by the NP electrospray as a function of NP concentration and spray time. After planting aged lettuce seeds in an unfavorable pH condition for germination (pH 5), treated lettuce seeds showed clear germination enhancement. Aged lettuce seeds (stored for over 10 months) in a MES buffer (pH = 5, a typical pH in acidic soils) had a natural germination percentage of ~ 40%. 

296
The germination of the same lettuce seeds reached its peak value (65%) when seeds were pretreated by electrospraying TiO$_2$ NP suspension (at the concentration of 1 g/L) for about 4 min. Prolonged NP electrospraying displayed less effectiveness on the seeds’ germination.

Figure 4b shows the germination of lettuce seeds when they were incubated in the buffer solutions of various pH values. Three sets of fresh lettuce seeds were used in each test incubation condition: untreated seeds (black bars), pre-treated seeds by electrospraying the solvent only (grey bars), and pre-treated seeds by electrospraying with TiO$_2$ NPs (white bars). When lettuce seeds were fresh (i.e., recently purchase from the seed company), the control seeds (seeds without pretreatment) usually had high germination percentages (~ 80%) even without NP electrospray treatment under a wide range of pH growth conditions (i.e., pH = 5~9, normal soil range). When the lettuce seeds were placed in an extreme harsh pH conditions (i.e., pH = 3), the control seeds showed minimal germination (~ 0%) while the NP-treated seeds recovered to a germination percentage of 20%. This result indicates that NP-treated plant seeds may be useful in to vegetate the polluted lands (e.g., phytoremediation of acid contaminated soil).

3.3 Effect of various NPs on the seed germination enhancement

To verify the feasibility of this proposed treatment, CuO and Au NPs were also applied in this study to pre-treat different sets of aged lettuce seeds. Figure 5a shows the seed germination results when aged lettuce seeds (stored for 16 months) were electrosprayed by CuO NP suspensions. Figure 4b gives the germination percentage when the same aged lettuce seeds were pre-treated by electrospraying Au NP suspensions at various concentrations. The germination percentages for the controls were also included in the figure as the reference. The results indicate that the electrospray of either CuO NP (1 g/L) or Au NP suspensions had significant enhancement effects on aged lettuce seeds. Similar to TiO$_2$ NP electrospray, the best seed
germination percentage for CuO NP electrospray were obtained in the case of a short spray time of 4 min. For the 4-min spray period, the best germination percentage occurred at the concentration of $10^7$ NPs/cm$^3$ for Au NP suspensions. Based on the above observation, nanomaterials of different compositions are possible to use for enhancing the seed germination.

We further performed the comparison test to characterize the effectiveness of various NPs on the germination of aged lettuce seeds in a prolonged incubation period (shown in Figure 6). Figure 6a shows the germination percentage under the electrospray treatment by suspensions of TiO$_2$, CuO and Au NPs. The results indicate that the treatment by electrospraying either TiO$_2$ or Au NP suspensions gave the most improved germination percentage. These pretreated seeds also germinated faster, while the control samples (seeds without pretreatment) showed a longer germination window (over seven days). Although the electrospray of the buffer solution (1 mM NaAc) accelerated the seed germination, we did not observe an improvement in the overall germination percentages in a seven-day incubation period. For agricultural and horticultural crops, delayed and sporadic germination is undesired because it reduces the harvest efficiency. NP-electrospray treated seeds may have the advantage of better crop productivity because of their early and homogeneous germination behavior.

No significant difference on the shoot length of germinated lettuce seeds among all treatment conditions (only the case with CuO NPs showing somewhat negative impact on the shoot length) was observed in this study (shown in Figure 6b). This observation suggests that the proposed treatment has no adverse effect on the early state of shoot development and could probably improve overall seed germination.

3.4 Mechanism for lettuce seed germination enhancement by TiO$_2$ NP electrospray
Plant seeds are incapable of germination at 100% even under the favorable conditions of temperature and hydration. In this study, we have demonstrated that the electrospray of NP suspensions can enhance the germination of lettuce seeds while immersing seeds in the buffer solution only (without spray) showed no significant impact on the seed germination percentage. Although the electrospray of buffer solutions without NPs showed minor positive enhancement on the seed germination, it was not as effective as the electrospray of NP suspensions. The possible explanation to the observed enhancement on seed germination due to NP electrospray treatment is the breaking of the coat-imposed seed dormancy. Electrosprayed NPs bombarding the seed coat may weaken the structure of the seed coat (Figure 3), which is considered as an influential factor in controlling seed dormancy (Gleiser et al. 2004).

In an aqueous suspension of NPs, it has been previously reported that NPs can slowly penetrate into seeds of various types and affect their metabolism in vivo (Navarro et al. 2008a; Ma et al. 2010b). For example, multiwalled carbon nanotubes (MWCNTs) were able to penetrate through the coat of tomato seeds after several days of co-incubation (Khodakovskaya et al. 2009). In such bulk solutions, NPs are often observed in the agglomerate form and their interaction with seeds is weak. In this study, the electrospray process effectively disperses most of NPs in singlet form, accelerates the velocity of NPs via the presence of electric field and space charge effect, and bombards the seeds by NP collision at high frequency. The proposed process thus increases the chance for NPs entering into seeds via piecing their coats or through natural pores. In case that nano-sized holes are created on the seed coat, oxygen transfer and water uptake might occur and drive the metabolic process for plant growth (Khodakovskaya et al. 2009).

Further, the effect of metal ions on seed germination can be excluded in our study as TiO$_2$ NPs are considered to be insoluble (< 5 ppb, confirmed by our ICP-MS measurement).
Because the incubation process for seed germination took place in the dark, the production of oxidative $\text{H}_2\text{O}_2$ through the reaction of light with TiO$_2$ NPs was minimized in this study. It may explain why toxicity of TiO$_2$ NPs, reported in the works of Menard et al. (2011) and Hund-Rinke and Simon (2006), was not observed in our investigation (Menard et al. 2011; Hund-Rinke and Simon 2006). For the case of CuO NP spray, the decrease in seed germination effectiveness is presumably because of the known phytotoxicity of CuO NP (e.g., release of Cu$^{2+}$ ions to interfere with seed functions) (Wang et al. 2010b; Baek and An 2011; Karlsson et al. 2008).

3.5 Shelf life of treated lettuce seeds

The shelf life test was performed on the lettuce seeds which had undergone the NP electrospray treatment, in order to evaluate the practical potential of the proposed method (Schwember and Bradford 2010). Aged lettuce seeds treated by NP electrospray were sealed and stored in the dark and at room temperature for various time periods (i.e., 1 day, 1 week, and 1 month) prior to the incubation. Figure 7 shows the germination percentage of aged and TiO$_2$ NP treated lettuce seeds after the defined storage periods. Significant enhancement on the germination of NP-treated seeds was observed when compared with the control. The germination percentage of aged seeds slightly dropped after one month storage, indicating that the seed coat of NP-treated seeds remained a sufficient protection for the seed embryo during the storage.

3.6 NP electrospray treatment of grass seeds

Several types of grass seeds (e.g., recalcitrant yarrow seeds and ragweed seeds) which have naturally low germination percentages under favorable growth conditions (typically 1~2% each year) were also used in this study to demonstrate broad applications of the proposed seed treatment. In this part of experiment, recalcitrant yarrow and ragweed seeds were subjected to 5 min electrospray of suspension with 1 g/L TiO$_2$ NPs. After culturing in de-ionized H$_2$O at pH = 7,
the germination percentages of recalcitrant yarrow seeds were enhanced from 1.6 (untreated) % to 6.8% (pre-treated) (shown in Figure 8). The germination percentage of ragweed seeds was improved from 1% (untreated) to 3% (treated). The germination percentage of grass seeds studied herein has the potential to be further improved by optimizing the electrospray operation conditions and NP concentration in spray suspensions. Because of the diversity of structure of seeds & coat of seeds in different species (Finch-Savage and Leubner-Metzger 2006), it may require different electrospray conditions to achieve the best result for various species as compared to that for lettuce seeds.

4. Conclusion

Electrospray of NP suspension was proposed to treat plant seeds for the enhancement of seed germination. A single-capillary electrospray system having a particle deposition stage was set up for this investigation. By applying a positive high voltage on the spray capillary and the electrically grounded stage, the electrospray was operated at the cone-jet mode for the production of monodisperse droplets. The solvent in droplets evaporated right after the droplet production, resulting in highly charged NPs in singlet form. Charged NPs were then accelerated in the presence of DC electrical field and space charge effect, and bombarded lettuce seeds on the deposition stage. Our study demonstrates that NP electrospray effectively dispersed NPs for breaking the seed coat to enhance seed germination. Our study further shows that the proposed treatment enables seeds to germinate under harsh environments (i.e., low pH soil). The enhancement on seed germination were also observed when electrospraying NP suspension of CuO and Au, independent of particle composition. Such treatment was further investigated and proven to be effective for certain weed seeds.
Our proposed seed treatment method can be further optimized by varying the NP concentrations/sizes, electric fields, and spray time. Such method can be cost effective for scaled-up for industrial applications because dilute NP suspensions were used in the treatment (estimate: 1 g of NPs can spray 3.6 million lettuce seeds). We believe that the proposed NP electrospray has the potential to be applied to various plant seeds (Pui and Chen 2000; Nadjafi et al. 2006). Meanwhile, environmental friendly NPs (Rieter et al. 2008; Yan et al. 2010) can be employed in the future to alleviate the concerns on the cytotoxicity of metal oxide NPs (Walser et al. 2012). In addition, the use of engineered NPs carrying DNA, plant hormone or other chemicals in the proposed process for seed treatment may open up new opportunities for broad application of nanotechnology in agricultural industry.
Acknowledgements

We appreciate valuable suggestions from Prof. G. Welbaum at Virginia Tech. We would also thank Prof. D. Ho from the Department of Biology at Washington University in St. Louis and Prof. I. Kong at Yeungnam University for close readings and helpful discussions of this manuscript.
References


El-Temsah, Y. S. and Joner, E. J. (2010). Impact of Fe and Ag nanoparticles on seed germination and differences in bioavailability during exposure in aqueous suspension and soil. *Environ Toxicol:* n/a-n/a.


Figure Captions

Figure 1  A schematic diagram of single-capillary electrospray setup with the particle deposition stage (used in this study).

Figure 2  The measurement of TiO$_2$ NP size distributions: (a) for a freshly prepared NP suspension (at 1 g/L TiO$_2$ NPs) measured by dynamic light scattering; (b) for gas-borne TiO$_2$ NPs after electrospray; (c) SEM image of freshly prepared NP solution (1 g/L) and (d) SEM image of TiO$_2$ NPs after electrospray.

Figure 3  SEM (a), TEM (b) and SEM-EDX (c, d) images of lettuce seeds treated by electrospray of TiO$_2$ NP suspension at a concentration of 1 g/L for 5 min. The SEM image (a) shows that TiO$_2$ NPs were individually adsorbed onto the seed surface. The TEM image (b) and SEM-EDX images (c, d) are for the cross sections of treated lettuce seeds. The images evidenced that the TiO$_2$ NPs can penetrate the coat of lettuce seeds and reside in the seeds. The scale bar in (b) indicates the length of 200 nm on the image.

Figure 4  Germination percentage of lettuce seeds after treated by TiO$_2$ NP electrospray as a function of spray time and NP mass concentration in spray solutions: (a) for the case with aged seeds (stored for 10 months) and incubated in buffer solution of pH 5; (b) for the cases with fresh seeds and incubated in the buffer solutions of pH = 3~11. Error bars in the figure are adapted from four replicates in each treatment.

Figure 5  Germination percentage of lettuce seeds after treated by NP electrospray: (a) for the cases with fresh seeds, using CuO NPs and incubated in the buffer solutions of pH = 7 for three days, and (b) for the cases with aged seeds (stored for 16 months), using Au NPs and incubated in the buffer solutions of pH = 7 for three days. Error bars in the figure are adapted from four replicates in each treatment.

Figure 6  Comparison of lettuce seed germination after the electrospray treatment using NPs of various materials (i.e., TiO$_2$, CuO and Au). Also included in the figure are the data for the control (without the treatment) and the case treated by electrospaying buffer solutions only for the reference. (a) germination percentage of seeds at both Day 3 and Day 7; (b) shoot length of seeds at Day 3 and Day 7; Error bar in (a) and (b) are adapted from four replicates in each treatment with aged seeds (Stored for 14 months and incubated in the buffer of pH = 5).
Figure 7  Germination of aged lettuce seeds (stored for 10 months) treated by TiO$_2$ NP electrospraying after being placed in the dark for one day, one week and one month prior to the incubation.

Figure 8  Germination of yarrow seeds after being treated by TiO$_2$ NP electrospray and incubated for 15 days. Also included in the figure are the germination of untreated seeds and those treated by spraying buffer solutions only (for the reference).
Figure 1
Figure 2

(a) 

(b) 

(c) 

(d)
Figure 3

[Images of microscopy images showing NPs and TiO₂-NPs]
Figure 4

(a)

(b)
Figure 5

(a)

(b)
Figure 6

(a) Germination percentage (%)

Control
Buffer spray
TiO$_2$ NP spray
CuO NP spray
Au NP spray

After 3 days
After 7 days

(b) Shoot length of germinated seeds (cm)

Control
Buffer spray
TiO$_2$ NP spray
CuO NP spray
Au NP spray

After 3 days
After 7 days
Note: * indicates a significant improvement in seed germination as compared with the control (based on Z-test, SE is based on four replicates).
Figure 8

![Bar graph showing germination percentage for different treatments: Control, Buffer spray, TiO₂ NP spray. The graph displays a clear increase in germination percentage from left to right.]
Vita

Gang Wu

Data of Birth        August 27, 1984

Place of Birth       Taizhou, China

Degrees             B.S. Biotechnology, June 2006

Degrees             M.S. Biological Systems Engineering, December 2010

Degrees             M.S. Energy, Environmental, and Chemical Engineering, May 2015