Modularity Analysis of Metabolic Networks Based on Shortest Retroactive Distances (ShReD)

A dissertation

submitted by:

Gautham Vivek Sridharan

In partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

in

Chemical Engineering

Tufts University

May, 2013

ADVISERS: Soha Hassoun, Ph.D., and Kyongbum Lee, Ph.D

To my grandmother, Rajalakshmi, who continues to dream that
I will win the Nobel Prize some day

My aunt Jayalakshmi, whose current struggle and determination to fight her cancer has inspired me to pursue an academic research career to study the disease

and

My mother, father, and sister

Acknowledgements

This is probably going to be longer than a typical thesis acknowledgements section. However, everyone I am about to mention has greatly impacted my success as a graduate student. When I reread this thesis years later, I want to be reminded of how much everyone's guidance and support has shaped the scientist and person I have now become.

Family

I begin with my extended family. Over the years, everyone has poured their love and excitement for my getting a Ph.D. Making everyone in my family proud of my accomplishments has been a huge source of motivation, especially my eldest uncle, Ramasubramanian, who sounded so excited on the phone when I told him I was close to finishing. In addition, my cousins Bharath, Shyam, and Anand have recently instilled in me the importance of staying in touch with both my heritage and language alongside anything I pursue in life. My being fluent in my mother tongue Tamil can be largely attributed to them and Anand reminds me of its importance because being able to think and reason in multiple languages enhances one's ability to learn and understand abstruse concepts. Finally, I would like to especially acknowledge my late uncle Madhavan, who was the first on my mom's side to complete a college degree and served as a model for future generations to pursue their education. His battle through cancer has inspired me to study the disease as an independent researcher some day.

I dedicate this thesis to my mother, father, and sister. I had the privilege of living with my parents for a couple of years during graduate school. Over the years, my mother has taught me the importance of maintaining relations with friends and family no matter how difficult work life becomes. My father, a scientist himself, has given me much advice over the years regarding technical research. However, the most important value he has instilled in me is that maintaining one's character and integrity is paramount and that being a good person will trump any other form of life success. I am so proud of my sister Preethy, who in many ways is far more mature than I was seven years ago. She currently aspires to go M.D., so I hope she can one day call me a 'fake doctor' because she has become a real one. However, she will be a champion regardless of what she ultimately chooses to do.

Friends

I have the best friends in the world. I would like to mention a few, who have taken extra efforts to provide support during my graduate career and ensure that I am always smiling and doing well.

Aruna Bhamidipati: She can relate to how I was raised the best, and is always able to provide advice and solutions to my problems while taking into consideration my core life values and beliefs.

Priyadarshani Giri: Darshani is a constant source of inspiration for being able to fight through the toughest of situations. Her advice is always simple yet profound. “Just get the tennis ball over the net,” she would say as I was losing to Nansen in a set of tennis.

Neha Kumar: NeGa always listens, and listens very well. She has also educated me deeply on many social issues and has forced me to think about topics in ways I never had before. She reminds me that being an educated individual involves a lot more than just scientific expertise.

Namrata Kothari: My sole Boston buddy from Cornell, we have had many dinners with lively conversations and topics of discussion/debate. She is always very genuinely interested in my research. She wanted to show up to my defense with a ‘D’ and ‘Fence’ sign, like they do at football games.

Avani Shah: Avani is a more passionate Patriots fan than I am, and a key member of the fantasy football league that I commission. The comedy and amusement that we have both shared from the events transpiring in the league have kept me constantly laughing every fall during graduate school.

Akrati Agarwal: Akrati was a senior at Tufts when I was a second year. She recruited me to join their senior dance as part of their TASA show, and introduced me to a lot of cool people who I continue to hang out with today.

Hrishikesha Vemaganti: I have known Rishi since I was 9 years old, and he has always had my back since. I was ecstatic when I found out he was moving to Boston in 2010. Ever since, he has spent countless hours and late nights in my lab waiting for me to finish experiments before we played tennis at night.

Nimil Sood: Nimil has followed the same path as me through high school, college, and now in pursuing a Ph.D. in (bio)chemical engineering. He is both a best friend and a scientific colleague and we have helped each other on our research over the years. We had a really amusing conversation one day when he asked me why I would ever multiply the Sherwood Number, Reynolds Number, and Diffusivity together. I was confused by the question, until I realized that’s what my published ShReD metric actually looked like.

Vaishal Patel: Vaish was my first true friend in high school and he followed me to Cornell as well. Conversations with him always involve dreaming big, and he has always pushed me to make sure I pursue greatness following my Ph.D.

Sean Sullivan: Sullys, another high school/college friend, is quantitatively the smartest person I know. Over Christmas break in 2010, he helped me write functions to construct the visual for the hierarchical tree of pie chart modules, which is frequently used in this thesis.

Nakul Paul: Naks has called me almost every day while I was in graduate school to share a joke or two, which always served as a good break. He is very good at rapid fire argument, and oftentimes used to joke that he was preparing me for my oral defense.

Xiao Li: Scott once convinced me to attend a game 7 Celtics playoff game against the Bulls during a week where I had a lot of work. I will never remember what I exactly had to do that week, but I will always remember that night at the Garden.

Nansen Yu: Nansen is my only other friend who I feel can match my intensity. Intensity in sports, and passion for anything we set out to accomplish.

Vinay Prabhu: Work hard, party hard. Thanks to Vinay, after I finished my oral defense, I thought to myself: “Boom”

Senthil Mudaliar: Senthil’s breadth and retention of knowledge in the sciences is indicative of how much he reads. He reminds us all to never stop reading, and to always stay on top of literature even if it is not directly related to one’s field. He’s going to make an excellent army physician.

Sharath Bhagavatula: Shay Shay, a phenomenal engineer who went M.D. aspires to join academic medicine. I hope to collaborate with him in the future as a colleague in academia.

Amit Bapat: Amit has taught me to avoid remaining in an academic bubble and instead develop a firm understanding of how markets work and how discoveries in the lab can be translated to products that are ultimately beneficial to humanity. We have many lively conversations about both science and finance.

Anandh Swaminathan: A fellow Ph.D. candidate from Acton, we chat all the time about theoretical systems research in biology. Only a second year at Cal Tech, he’s already way far ahead in research than I was at his age.

Collaborators

Ehsan Ullah: Ehsan, a graduate student in Prof. Hassoun’s group in computer science, directly contributed to the work in Chapter 3 of this thesis. After the completion of this thesis, we will be submitting the chapter as a manuscript for publication together. We would listen to A.R Rahman music while we coded together.

Mona Yousofshahi: Mona, another graduate student in Prof. Hassoun’s group has been a great resource for coding help. She developed the probabilistic path search algorithm referenced in Chapter 5 of this thesis.

Kyungoh Choi: Kyungoh, a graduate student in Prof. Jayaraman’s group at Texas A&M contributed to work presented in Chapter 5 of this thesis. She prepared the mice samples, and conducted the *in vitro* activity assay tests.

Kyle Quinn, Ph.D.: Kyle and I have worked together to develop correlations between an optical redox ratio measured by fluorescence microscopy and biochemical cofactor measurements using LC/MS-MS.

Former Lee Group Graduates

Hai Shi, Ph.D., Ning Lai, Ph.D.., Ryan Nolan, Ph.D., Kate Carson: Lee lab members senior to me taught me laboratory as well as modeling techniques. They brought me up to speed really quickly when I joined the group.

Hana Sheikh: She worked on a very interesting problem on identifying significant parameters in a metabolic kinetic model. We've had many lively discussions on the topic. A frequently brought up joke was that she was a senior when I started graduate school, then went to work for a couple of years, and came back to finish a masters while I still remained in Sci Tech.

Michael Yi: It was a great privilege to mentor Mike, an undergraduate researcher in our group. He directly contributed to the fat models used in Chapter 4 of this thesis. He's doing big things at Stanford now as a graduate student.

Current Lee Group Students

James Sims, Sara Manteiga, Julie Paul, Prity Bengani, Venkatesh Gopi, Charmian Wu, Long Bin Pan, Kim Stachenfeld, Brian Rohr, Rob Dimatteo: The current Lee group has been very supportive of me over the years. It is especially refreshing to see how passionate the newer students are to learn quickly and conduct research. The Lee laboratory seems to bring in only the best graduate and undergraduate students.

SciTech Friends and Colleagues

Brett Boghigian, Ph.D. His publication record while he was a graduate student here served as a huge source of inspiration for me. I aspired to become the 'next Brett' of the department. When I first came to Tufts, Brett and I immediately recognized each other in that we had played tennis on opposing teams in high school, being at rival schools. When we play tennis now, we spend a lot of time warming up volleying at the net so that we could talk about research and beyond.

22 Warner St: George Cladaras, Matthew Rutter, Timothy Lawton: It was a pleasure coming home every day to great group of guys during the years I lived there. I am looking forward to all of you finishing school soon.

Mathew Boucher, Ph.D. Matt, my best friend from graduate school and member of 22 Warner, is the best researcher I have met so far. He is meticulous, thorough, well read, and churns out high impact manuscripts at a rate that may not seem humanly possible. We have had many great scientific discussions at Soundbites, our favorite lunch spot. He is taking a few years to try out the private sector, but I do hope that he returns to pursue academia since I strongly believe that those who can, should.

Committee/ChBE Faculty

Christos Georgakis, Ph.D.: Professor Georgakis's class on design of experiments has really shaped the way I think about quantitative modeling. We've had many conversations on parameter estimation and nonlinear optimization. I hope to write a paper with Prof. Georgakis some day.

Steve Matson, Ph.D.: Prof. Matson has been of great help on my committee. He always has a 30,000 foot view of the problem and is able to dissect the merits and shortcomings of a study from a practical application perspective. He suggests I think about how the graph theoretical work I use for metabolic modeling can be used in other fields and applications.

Arul Jayaraman, Ph.D.: Prof. Jayaraman spending his sabbatical here at Tufts was a game changer. I was immediately engrossed in the project he proposed on measuring and quantifying gut microbiota-derived metabolites *in vivo*. I felt as though I had a third advisor guiding me to the finish line. We also connected well, both being ‘real Indians’.

Advisors

I could write another complete dissertation on just how much Dr. Soha Hassoun and Dr. Kyongbum Lee mean to me and how they have shaped me over the years. For the sake of some brevity, I will just list a few points.

Soha Hassoun, Ph.D.: Prof. Hassoun has an incredible ability to take abstract ideas and provide a methodogical strategy for translating those ideas into solvable problems with organized action items. In the field of theoretical sytems biology, when it is often tempting to generalize based on a few empirical observations, she has taught me to show restraint and not make such generalizations without quantitatively rigorous proof. The amount of biology she has learned in the five years I have witnessed is amazingly impressive. When I become an academic some day, I hope to be as brave as her by diving into a new field unfamiliar to me to pursue novel collaborations

Kyongbum Lee, Ph.D.: Prof. Lee is the ideal advisor and mentor in every sense. His vast breadth of knowledge in biology, chemistry, chemical engineering, and computer science allows him to make unique connections to develop truly innovative ideas that span multiple traditional disciplines. I have knocked on his door after business hours almost every day for 5 years and to this day I am surprised he has never told me to leave because he was too busy. When I am a professor, I hope to be as available to my students for mentoring. Even when we meet, much of our conversations surprisingly had little to do with ShReD or modularity. We would discuss sports, finance, and sometimes politics. During each one of these conversations, Prof. Lee would introduce some new concept or idea I had not thought of before, prompting me to look it up immediately after to learn more about it. When I first joined graduate school, he told me that the wave equations from quantum mechanics resemble the Black-Scholes derivative pricing model. Piquing my interest, I learned more about it and today options trading and modeling their pricing has become a side hobby of mine. With Professor Lee, I felt as though I received a complete education on a broad range of subjects.

Soha Hassoun, Ph.D. & Kyongbum Lee, Ph.D. together: I also need to acknowledge the two of them together as a collaborative team. They have conducted very fruitful journal club meetings where the chemical engineering and computer science students could learn from each other. Soon after participating, I was comfortable talking about graph theory, and algorithms such as ‘depth first search’, while the CS students were able to articulate the difference between equilibrium and steady state. In an age where individual accomplishments seem to be emphasized in academia, I believe that Prof. Lee and Prof. Hassoun’s selfless attitude toward research deserves special recognition. Together, they epitomize the interdisciplinary research that Tufts preaches as a university.

Funding

NSF award number 0829899

A few quotations that guide me

“Arise, awake, and stop not till the goal is reached”

– Swami Vivekananda

कर्मणयेवाधिकारस्ते मा फलेषु कदाचन।
मा कर्मफलहेतुर्भूर्मा ते सङ्‌गोऽस्त्वकर्मणि।

(Let right deeds be thy motive, not the fruit which comes from them)

– Bhagavad Gita, Chapter 2

எல்லா புகழும் இறைவனுக்கே!

(all praise is dedicated to God)

– a humble A.R. Rahman after winning the Oscars.

Abstract

Cellular metabolism is very complex. Large scale networks that are used for modeling single-cell organism or tissue-specific systems typically comprise of several thousand reactions, each representing a unique biochemical conversion of substrate to product. These *in silico* models have the potential for predicting how a cell may respond to a perturbation in the form of either a genetic intervention or external stimulus. However, the sheer complexity of these networks remains an impediment for the construction of predictive kinetic ODE models, because the number of system parameters that need to be estimated typically far exceeds the available experimental data and most estimated parameters are not statistically identifiable. Alternatively, graph-based modeling of metabolic networks, where reactions can be denoted by nodes and their interactions described by directed edges, allow one to survey solely the topology of the network and identify structural features that may offer predictable dynamics. Moreover, graph theoretical tools allow for the discovery of modules, or a subset of reactions containing few inputs and outputs, that together function in concert to isolate perturbations from propagating to the rest of the network, a characteristic of metabolic robustness. In this regard, the systematic modularity analysis serves to reduce the complexity of metabolic models and identify modules that both confer robustness and reveal strong coupling among reactions that may not necessarily be intuitive by viewing a two-dimensional cartography of metabolism.

In this thesis, the governing hypothesis is that retroactive, or cyclical, interactions in the form of feedback loops and metabolic cycles engender robustness, and serve as a defining structural feature for the systematic identification of functional modules. As such, a graph-theoretical metric called the Shortest Retroactive Distance (ShReD) is introduced to be used in

conjunction with a known network partition algorithm to produce a hierarchical tree of modules, each enriched in cyclical pathways and allosteric feedback loops. Applied to a hepatocyte (liver cell) metabolic network, the ShReD-based partition identifies a ‘redox’ module that couples reactions from apparently distant pathways such as glucose, pyruvate, lipid, and drug metabolism through the shared production and consumption of NADPH, suggesting that cofactors greatly influence the modularity of the network. Recognizing that metabolic networks are not static, a metabolic flux-based edge weighting scheme is proposed to capture the relative engagement between reaction nodes in the graph network. Applying the ShReD-based partition algorithm to weighted adipocyte (fat cell) networks reveals that major physiological changes such as cellular differentiation lead to substantial reorganization in the modularity of the network. In addition, ShReD-based modularity serves as a platform for a targeted motif search within functional modules to discover novel metabolic substrate cycles (a.k.a. futile cycles), which have been recently proposed to be targets for obesity and even cancer. Identifying these substrate cycles requires elementary flux modes (EFM) computation, which would otherwise be infeasible on a large scale network.

Prospectively, modularity analysis of metabolic networks provides theoretical guidance for which reaction rates and metabolite levels may be altered in the face of a perturbation. To experimentally confirm predictions, targeted metabolomics using tandem mass spectrometry (LC/MS-MS) is used to obtain absolute quantification of metabolite concentrations. As an example, an *in silico* model predicts a set of tryptophan-derived metabolites that can only be exclusively produced by the gut microbiome and may have anti-inflammatory properties. *In vivo* levels of these indole-backbone metabolite levels are quantified in cecum samples from mice at two different age groups. Statistically significant differences between the two groups suggest that age influences the microbiome composition as well as the metabolites they produce.

Table of Contents

List of Figures

List of Tables

1. Introduction: Modularity and Robustness in Metabolic Systems

1.1 Investigating Metabolic Perturbations

Cellular metabolism is a complex dynamic process describing all chemical transformations of organic matter in living systems. For single cell organisms, such as bacterial *E.coli* or yeast *S.cerevisae*, the functional objective of the cell may be to maximize its growth and proliferate (Varma and B. O. Palsson, 1994). The cellular objective for higher order mammalian cells however may be not be as clear, since tissue specific cells participate in many unique functions to maintain organism-wide homeostasis (R. P. Nolan *et al.*, 2006). In either case, the ability to predict how cells respond to metabolic perturbations would be invaluable in the context of both engineering microbes producing desirable chemicals or studying the metabolic effects of drug treatments. There are many types of perturbations that globally affect metabolism and they can be defined as any form of external intervention or stimulus that affects a cell's nominal steady state. For example, genetic perturbations involve the knockout of a particular gene, which can drastically affect an organism's metabolism compared to the nominal wild type case (Siddiquee *et al.*, 2004). A transcriptional perturbation, as the name suggests, would be characterized as an external agent that acts to either activate or inhibit the transcription of certain genes by binding to transcriptional factors (Qin *et al.*, 2007). In addition, a perturbing chemical agent may also directly bind allosterically and alter the activity of an enzyme, affecting both metabolic (Si *et al.*, 2009) as well as signaling pathways (Yun *et al.*, 2008). Finally, perturbations can include drastic shifts in the environment the cells experience, such as temperature shifts (R. P. Nolan and Lee, 2010), or sudden accumulation or depletion of available substrates (Hoppe *et al.*, 2007).

Studying the impact of perturbations on cellular metabolism offers many practical applications. For decades, metabolic engineers have studied the impact of knocking out or up-regulating genes to improve the production and secretion of desired compounds, both small molecules (Boghigian *et al.*, 2010) as well as larger proteins in *E.coli* (Goeddel *et al.*, 1979). However, even the most rational approaches to genetic modifications have produced counterintuitive results. For example, the knockout of a gene, intended to target a single pathway, may have a lethal impact by globally affecting the function of many reactions essential to the cell's survival. Those reactions may draw from the pool of metabolic cofactors, such as ATP, NADH, or NADPH, and can broadly affect reactions in parts of metabolism conventionally thought to be distant from the point of perturbation (Blank *et al.*, 2005). Even more surprising, a genetic intervention may have little or no impact on metabolism, as several enzymes may catalyze the same conversion of substrates to products (L. Zhu *et al.*, 2012), which is known as redundancy. That biological systems exhibit both robustness as well as extreme fragility to certain perturbations is widely discussed in literature (Quinton-Tulloch *et al.*, 2013a).

The challenges involved in predicting the impacts of metabolic interventions are also relevant to mammalian systems, especially in the context drug targeting (Kitano, 2007). Killing pathogenic organisms, for example, requires targeting of fragile points. The pathogen *Vibrio cholerae*, the cause of cholera disease, can be treated using small-molecule inhibitor (virstatin) which acts on a transcriptional regulator ToxT that is unique to the organism (Hung *et al.*, 2005). On the other hand, treatment of certain systemic disorders requires local targeting of specific enzymes while minimizing global impacts on metabolism causing adverse side effects. One such disorder, the metabolic syndrome, characterized by hypertriglyceridemia, hypertension, high fasting glucose, and low high-density lipoprotein (HDL) is a nationally trending epidemic where insulin resistance can drastically alter sugar and fat metabolism to the extremes of type II diabetes and obesity (Parekh and Anania, 2007). Treatments have included thiazolidinediones, such as troglitazone, that act on a transcriptional factor PPAR-gamma, but severe hepatotoxicity induced by xenobiotic transformation in the liver prompted its withdrawal from the market (M. T. Smith, 2003). Recently, enzyme targeted drug interventions that alter liver cell (hepatocyte) and fat cell (adipocyte) metabolism have been proposed to provide a desired outcome such as reduced triglyceride synthesis and storage (Si *et al.*, 2009). In this regard, siRNA mediated knockdown of specific genes by attacking mRNA fragments has gained increasing attention (Wilcox *et al.*, 2006), and several companies have siRNA-based therapeutics as part of their pipelines to treat disorders at the level of metabolism (Whitehead *et al.*, 2009).

For the engineering and therapeutic applications discussed thus far, the importance of predicting metabolic outcomes is evident. Designing and implementing biological experiments to test the impact of such perturbations are time consuming and expensive. As such, quantitative models that can predict cellular behavior at various metabolic states would help identify more effective metabolic targets, as well as explain why some perturbations yield counterintuitive results. However, metabolic systems are very complex. The complexity of metabolism rises from the interplay of thousands of enzymes catalyzing the conversion of substrates to product, each with its own transcriptional and allosteric regulatory controls. The ability to map these processes has been greatly augmented by the genomics and transcriptomics revolution, which enable one to determine which genes are expressed in specific cell types. Meticulous interpretation of the data has led to the construction of large genome-scale databases cataloguing metabolism (KEGG)

(Minoru Kanehisa *et al.*, 2010, 2006) which greatly aid in model curation. These model networks can then provide insight as to whether or not a perturbation would have a localized impact on metabolism, or a more drastic effect on parts of the network previously thought to be distant. Indeed, the relative 'distance' between certain reactions and the source of the perturbation is not always obvious, and conventional two-dimensional cartography of metabolism can be misleading at times. In this regard, novel quantitative modeling approaches enhance our understanding of how a perturbation propagates throughout cellular metabolism, and hopefully serve to reduce the number of costly experiments one has to perform in the lab.

1.2 Modularity and Robustness

1.2.1 Modularity

A metabolic network can exhibit *modularity,* in that a subset of reactions and metabolites exhibit distinct functionality separable from other modules (Hartwell *et al.*, 1999). Intuitively, metabolic modules should be semi-autonomous subgroups of reactions that have stronger interactions among themselves while exerting a weakened influence on the rest of the system, thus exhibiting minimum crosstalk with other modules. The compartmental nature of higher order eukaryotic systems offers an intuitive view of modularity. For example, a perturbation to a reaction taking place in the endoplasmic reticulum may not immediately impact reactions in the mitochondria. Moreover, it is also conceivable that subsets of reactions even across multiple compartments, if regulated a certain way, can function autonomously and isolate external perturbations from propagating to other modules. Kitano has qualitatively described this as biological *robustness*, and suggested the modularity of metabolic systems promotes this very robustness (Kitano, 2004). As such, the modularity of a metabolic network is directly related to robustness. Since robustness is used in many different contexts (Larhlimi *et al.*, 2011), we seek to define local and global robustness in response to a perturbation within a dynamical system context.

1.2.2 System Context

Cellular metabolism can be mathematically modeled as a dynamical system that exchanges energy and mater with its environment. The state space is characterized by variables representing metabolite concentrations and reaction rates. A *Steady state* refers to any state in which metabolite concentrations do not uncontrollably accumulate or deplete to the point that it is toxic to the cell. This can be described mathematically in the case where a metabolic dynamic system features an asymptotically stable steady state, which occurs when the real parts of the linearly approximated Jacobian's eigenvalues are all negative (Variano and Lipson, 2004). One system property is homeostasis, which is the tendency of the system to remain in or near the *current* steady state. A steady state is considered *stable* if minor fluctuations in the state variables do not cause great deviation from the reference steady state. Feedback, a process in which downstream information can be used to modify the state upstream, is essential in controlling and regulating the state space. Positive feedback loops allow transitions between two different stable states of a system, while negative feedback loops can maintain a downstream variable within a narrow range, despite widely fluctuating upstream activities. The effects of such feedback loops may be local or global within the system.

1.2.3 Local vs. Global Robustness

For many engineering applications, the ability to maintain a steady state in the face of a perturbation is an important functional outcome. We refer to the ability of the system to maintain its current steady state or reach a new stable steady state due to perturbations as *global* *robustness.* Global robustness hence does not necessarily imply tendency towards homeostasis. While the terms stability and robustness are often used interchangeably in literature, a stable system need not necessarily be robust to perturbations, since the perturbation can lead the system towards toxicity.

An alternate desirable functional outcome may be *local robustness,* where a perturbation causes changes within the state but *only* in parts of the subsystem. Kitano's biological robustness certainly encompasses our local robustness definition. Equivalently, we can define local robustness in terms of metabolic control analysis using flux control coefficients that measure the local response of an enzyme to changes in its environment (Fell, 1992). It follows that local robustness is thus a state-dependent property, and that system-wide global robustness is a necessary condition for local robustness but local robustness is not a guaranteed consequence of global robustness. Conversely, a subset of state variables (i.e. flux rates) being locally robust to a perturbation does not imply global robustness, as some metabolites may never reach a new steady state concentration. Our goal is to ultimately discover functional modules that confer local robustness, in that reaction fluxes within the perturbed module should be altered more than those outside the module. Performing *in silico* experiments on detailed kinetic metabolic models would enable the discovery of this modularity by computing the flux control coefficients for each enzyme-reaction pair.

1.2.4 Analyzing Modularity Using Dynamic Models

Indeed, the holy grail of mathematical modeling of metabolism as a system is to ultimately have organism or tissue-specific predictive dynamic models. These models are typically constructed as a system of ordinary differential equations (ODE) describing the change in concentration of metabolites as a function of time (Chalhoub *et al.*, 2007). The system's metabolite concentration changes are governed by the rates of reactions that produce and consume the metabolites, which typically have highly nonlinear expressions based on enzyme kinetics. Michaelis-Menten type saturation kinetics are most commonly used, where the Vmax parameter represents the maximum attainable rate of the reaction (accounting for amount of enzyme) and the saturation Km parameter is a measure of the affinity the substrate has for the enzyme. Allosteric regulatory interactions greatly add to the complexity of the reaction expressions. Once the kinetic model is assembled, *in silico* experiments can be conducted to test the effect of perturbations such as gene knockouts, or enzyme knockdowns /inhibitions on metabolic fluxes by running simulations after varying relevant parameter values (e.g. reduced Vmax values to denote a knockdown or inhibition (R. P. Nolan and Lee, 2012).

Unfortunately, despite ambitious strategies (Jamshidi and B. Ø. Palsson, 2008), the development of such large scale kinetic models remains a formidable challenge due to the number of parameters that have to be estimated for the model with limited data on metabolite transients (Maier *et al.*, 2010). Some have tried to piece together kinetic models with parameters measured from *in vitro* enzyme kinetic experiments. However, the predictive power of these models is limited, presumably due partially to the fact that data collected *in vitro* may not reflect the *in vivo* environment (Alves *et al.*, 2008). Recently, Nolan and Lee developed a predictive kinetic model to describe CHO cell metabolism, but only featured dynamics for a small set of exchange reaction fluxes, and used constrained-based modeling to determine intracellular fluxes, a technique referred to as dynamic flux balance analysis (R. P. Nolan and Lee, 2010, 2012). Such reduced models with lumped reaction kinetics are useful for simulating a subset of metabolism that accounts for most of the carbon flux, but are limited for understanding the detailed influences of individual reactions.

While this is somewhat discouraging, the question we are seeking to answer for a metabolic system: “What affects what?”, i.e. which reaction rates are significantly altered as a result of a perturbation and which less affected, need not necessarily require such a rigorously quantitative approach with kinetic models. Alternatively, structural or graph-based modeling of metabolism offers a semi-quantitative modeling platform to survey the connectivity of reactions and metabolites in a metabolic network. Such an approach allows one to investigate structural features of metabolism that may correspond to predictable dynamics (Prill *et al.*, 2005). Moreover, the modularity of a metabolic network, determined based on topological structural features may offer predictable dynamics with regards to the system’s functional modularity and local robustness.

1.3 Identifying Modularity Using Graph-Based Modeling

A metabolic network graph comprises component nodes, typically either reactions or metabolites, and edges linking the nodes, denoting their connectivity relationships. Graph networks can be represented as metabolite-centric, where metabolites are treated as nodes, and the reactions that link a substrate to product are denoted as edges (Yoon *et al.*, 2007). Alternatively, graphs can be represented as reaction-centric, where nodes refer to reactions or enzymes, and edges denote connectivity between reactions based on the production and consumption of intermediary metabolites (Ma *et al.*, 2004). Metabolic graphs can also be bipartite, with two sets of nodes, both reaction and metabolite, and a link from a metabolite node to a reaction node if the latter consumes the particular metabolite, or a link from a reaction node to a metabolite node if the latter is produced by the reaction (Ma'ayan, 2009). There is no universally accepted way of representing a metabolic network, as it depends on the modeling application. Metabolite-centric graphs have been considered the intuitive abstraction, as reactions represent flow of material, denoted by the edge. On the other hand, reaction-centric graphs focus on enzymes, the functional unit of metabolism.

Representation of metabolic networks as graphs have allowed for the discovery of certain characteristic global topological properties, such as small world, scale-free organization, and structural modularity. The small world property suggests that despite networks being very complex, the average path length between components is surprisingly short, due in part to many hub metabolites that participate in many reactions. It is also related to the scale-free nature of metabolic networks, where the degree distribution (fraction of nodes with k connections, or degrees) is shown to follow a power law (H Jeong *et al.*, 2000), albeit a contested claim (Goemann *et al.*, 2011). Graph theory offers efficient algorithms for performing calculations such as shortest path (Floyd, 1962), and clustering coefficient (Ravasz *et al.*, 2002). Several comprehensive reviews on how to represent networks as graphs and perform these calculations are available (Montañez *et al.*, 2010).

Several investigators have already looked into using graph-based tools to discover the modularity of metabolic networks (Ederer *et al.*, 2003; Holme *et al.*, 2003; Ma *et al.*, 2004; Papin *et al.*, 2004; S Schuster *et al.*, 2002; J. Zhao *et al.*, 2006). Ravasz and coworkers published a seminal paper in the field, reporting that scale-free networks with hub metabolites can also possess modularity if hierarchically organized (Ravasz *et al.*, 2002). More recently, hierarchical modularity has been associated with enhanced global robustness to perturbations, by demonstrating that random attacks to a more modular linear dynamic system led to more asymptotically stable solutions (Variano and Lipson, 2004). Various methodologies have been employed for systematically uncovering the hierarchical modularity of metabolic networks, which usually involve coupling the identification of a structural feature to a clustering or partition algorithm. For example, Ravasz investigated the *topological overlap metric* based on connectivity to obtain distances between nodes, and then used an average linkage clustering routine to obtain the hierarchical organization (Ravasz *et al.*, 2002). Newman developed a metric that computed the difference between the actual and expected number of connections between two nodes and used a binary partition algorithm that maximizes a modularity score Q at each iteration (M. Newman and Girvan, 2004; M. E. J. Newman, 2006). A useful consequence of such top-down partitioning is that it enables the structural metric to be re-computed for each sub-network in the hierarchy before a partition decision is made.

1.4 Cyclical Interactions Confer both Local and Global Metabolic Robustness

While the approaches thus far have successfully identified hierarchically organized structural modularity in metabolic networks, they have relied solely on *local* connectivity-based metrics. It follows that one can look for other motifs or structural features that confer robustness. In addition to facilitating robustness, the uncovered modules should also in principle uncover strong coupling between certain reactions. In this thesis, the governing hypothesis is that cyclical interactions help identify reactions that mutually influence each other and strongly affect each others' reaction rates in the face of a perturbation, but also capture motifs that confer local robustness.

Kitano and others have long qualitatively described the role feedback control plays in maintaining cellular functions in the face of a perturbation (Kitano, 2004). Moreover, metabolic cycles are also gaining attention as a mechanism to retain substrates and contribute to system stability and robustness (Kritz *et al.*, 2010). To provide a more quantitative evidence for these claims of system robustness, structural kinetic modeling (SKM) (Steuer *et al.*, 2006), a recently proposed approach, has been used in the absence of detailed kinetic models of real systems. SKM takes into account both the stoichiometry as well as the dynamics of a locally linear approximation of the system at a particular steady state. The parameters of the system, which are represented as normalized degrees of saturation of a reaction with respect to a substrate, are then sampled at random. The result is an ensemble of kinetic models about a given steady state, and the stability of each model can be evaluated by the Jacobian matrix. A metabolic system here, whose structure is defined solely by the stoichiometry, is said to be more robust if a greater fraction of Jacobians from the ensemble denote asymptotically stable steady states (Steuer *et al.*, 2007; Reznik and Segre, 2010; Grimbs *et al.*, 2007; Steuer *et al.*, 2006). That is, the system is *more likely* to reach a stable steady state for a broad range of parameter values, implying greater global robustness to reaction perturbations denoted by altered parameter values. Using the Structural Kinetic Modeling (SKM) approach, Grimbs and coworkers report more quantitatively that metabolic systems with allosteric regulatory information embedded exhibit global robustness (Grimbs *et al.*, 2007). Similarly, Reznik and Segre extended SKM to combine analytical and computational approaches to show that single input, single output metabolic cycles are always stable for a wide range of conditions tested (Reznik and Segre, 2010).

One can also show that these cyclical interactions may contribute to local robustness. To illustrate the impact of allosteric regulation, we consider a toy metabolic system, modeled by ODEs shown in Figure 1-1.

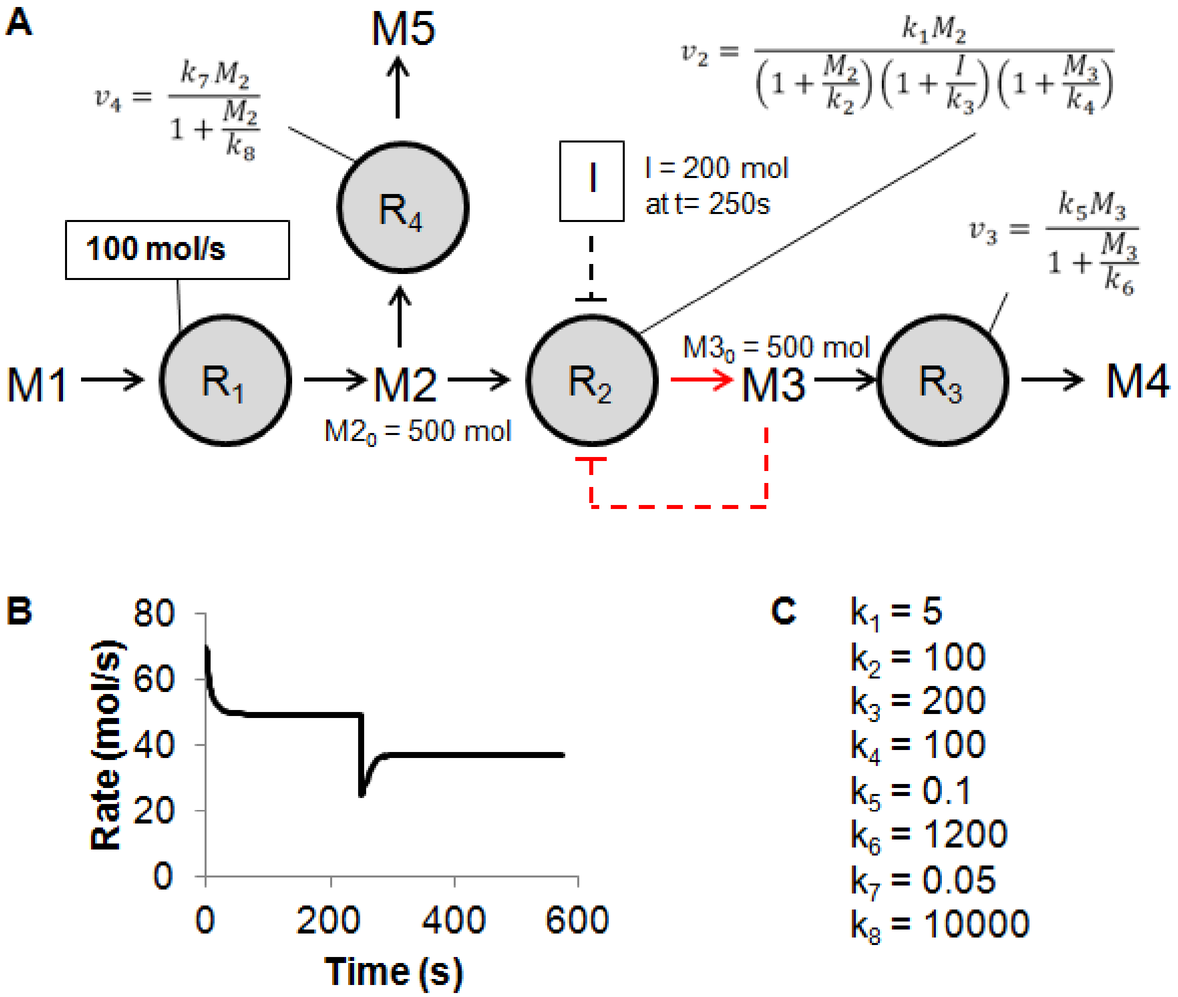


Figure 1-1: Allosteric regulation attenuates impact of perturbations

(A) Model kinetic reaction system with a fixed input rate R_1 at 100 mol/s and rate expressions for R_2, R_3, and R_4. (B) Rate of reaction 2 in mol/s as a function of time. The perturbation is caused by inhibitor I at t = 250s. (C) The values for the rate expression parameters.

An input of 100 mol/min (use of molar units is arbitrary) of flux is split at a branch point and is distributed between producing M5 and M4. An external agent, I, acts to allosterically inhibit R2's reaction rate, as does the product of the reaction, M3. In this specific example, the inhibitor I acts to immediately drop R2's reaction rate by 50%, but the system is able to partially compensate and recover the loss such that R2 only experiences at 25% reduction in flux once new steady state is reached. The accumulation of M2 post-perturbation does not explain the recovery in this case because the M2 concentration is in the saturated regime with respect to R2's reaction rate and any increase in reaction rate that would be driven by the k_3M2 term is offset by the M2/k_2 term in the denominator of R2's kinetic expression. On the other hand, with the immediate reduction in R2's reaction rate, M3 depletes from ~800 moles to 550 moles, which alleviates the allosteric binding of the metabolite on the enzyme and accounts for the 25% recovery in reaction rate. It is very important to note that not all systems with this structural motif will behave in this manner, as system dynamics are both parameter specific and state dependent. The intent here is more to show feasibility, or proof of principle, that allosteric inhibition can serve to locally attenuate a perturbation.

Similarly, the metabolic substrate cycle motif can also serve to isolate a perturbation and promote local robustness. For example, a similar toy model shown in Figure 1-2 involves the same branching as before, except a substrate cycles shuttles M3 back to M2 at the expense of a reduced cofactor.

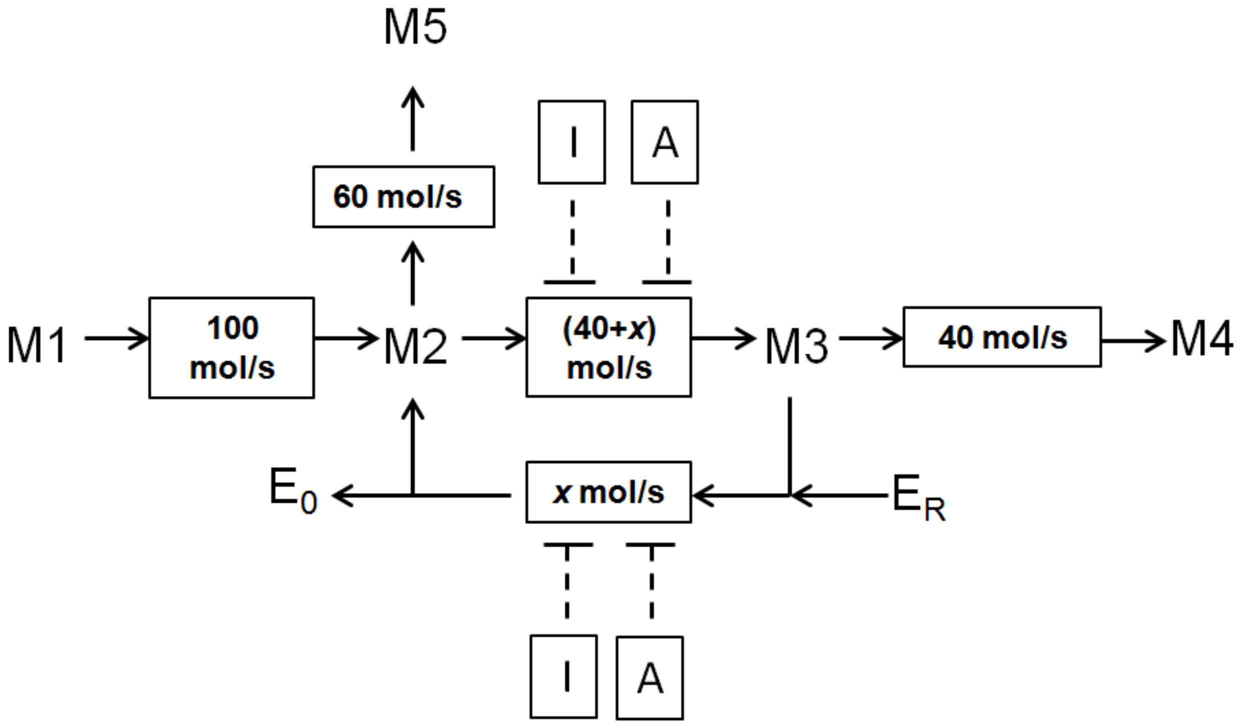


Figure 1-2: Model substrate cycle

With this system, one can show that a perturbation to R2 caused by inhibitor I, can be isolated to the substrate cycle if the same agent, I, also inhibits R4. That is, once at the new steady state post-perturbation, R2 and R4's fluxes can both decrease by the same amount without altering the output fluxes R3 and R5. Once again, this behavior is not indicative of all substrate cycles. In fact, the reactions in the substrate cycle involving phosphofructokinase I and fructose 6 bisphosphatase are regulated by AMP in opposite directions. That is, AMP inhibits FBPase I and activates PFK I, so a perturbation in the form of more available AMP would not be isolated by the substrate cycle, but rather would act as a switch for promoting glycolysis and shutting down gluconeogenesis. Similar to the example on allosteric regulation, we are simply highlighting that this motif is in theory capable of local robustness in certain specific cases, defined by the parameters of the system.

1.5 Modularity Based on Cyclical Interactions

Since there is evidence to suggest that cyclical interactions in the form of substrate metabolic cycles and allosteric feedback contribute to dynamic robustness, it follows that looking for these structural features using graph-based modularity detection algorithm may uncover functional modularity. Recently, Saez- Rodriguez and coworkers explored retroactivity, a concept borrowed from systems theory that describes the effect of a downstream element on an upstream one (Saezrodriguez *et al.*, 2005), in the context of modularity analysis. In their work, retroactive connections were established if neighboring components mutually influenced each other, either directly based on stoichiometry, or indirectly based on feedback. They hypothesized that the ideal modularity of a network would be one where the retroactive connections between modules, or intermodular cross-talk would be minimized (Saez-Rodriguez *et al.*, 2008), and partitioned networks using Newman's modularity score (M. E. J. Newman, 2006). However, these retroactive connections are solely based on nearest-neighbor interactions and do not account for longer range cyclical interactions and feedback loops. In fact, many allosteric feedback mechanisms involve the inhibition of a reaction by a compound produced by a reaction several reaction steps downstream. In addition, while the example shown in Figure 1-2 involved a two-reaction substrate metabolic cycle, there are examples of substrate cycles that span multiple reaction steps. For example, Gebauer and coworkers illustrate a 13 step substrate cycle that involves glycerophospholipid and pyrimidine metabolism with the overall consumption of ATP (Gebauer *et al.*, 2012).

In this light, the objective of this thesis is to extend the work of Saez-Rodriguez and coworkers by developing a graph-based metric to account for these more distant cyclical interactions. We propose a novel metric, ShReD (Shortest Retroactive Distance), which denotes the length of the shortest directed cycle that incorporates the two reaction nodes (Sridharan *et al.*, 2011). Similar to Saez-Rodriguez et al, we hypothesize that the ideal modularity would be one that minimizes cyclical interactions that span multiple modules. Analogous to Newman, who computed difference between the actual number of connections and the expected number of connections between two nodes, we say that if a reaction pair has a shorter ShReD than expected, then they should belong to the same module. Combining this metric with Newman's partition algorithm, the goal is to obtain a hierarchy of modules, each with a subset of reactions that feature feedback loops and metabolic cycles. In this introduction, we have thus far shown that these motifs confer local robustness, and modules with these motifs should in principle serve to isolate external perturbations. This thesis will now expand on the details of how to identify these ShReD-based modules how they reveal complex interactions among reactions, and provide some guidelines on how to experimentally monitor changes in metabolism. The following outline highlights the flow of the thesis.

1.6 Thesis Outline

Chapter 2

We formalize the definition of the ShReD metric as a graph-based tool to capture distant cyclical interactions in a metabolic network. We develop a systematic algorithm for obtaining hierarchically organized modules of reactions. Finally, we test the algorithm on both an EGFR signaling model as well as a liver metabolic network, featuring reactions for the drug metabolism of troglitazone.

Chapter 3

To determine if ShReD-based modules do indeed possess features that isolate perturbations, we perform a targetd motif search on modules obtained by partitioning hepatonet1, a large scale hepatocyte model. We look for cyclical elementary flux modes (EFM) to identify possible substrate cycles within modules.

Chapter 4

We seek to improve the ShReD-based modularity algorithm by weighting edges based on metabolic flux data. This way the metabolic network is not treated as static, but rather dynamic depending on the metabolic state of the cell. We develop a generally applicable metabolic flux-based edge weighting scheme for reaction-centric graphs and apply ShReD-based partitioning on the adipocyte reaction network at multiple metabolic states, each with unique flux distributions. This allows for the comparison of modularity across different metabolic states for the same base network.

Chapter 5

To experimentally determine the impact of perturbations to metabolic modules, one needs to measure changes in metabolite concentrations before and after the perturbation. With a targeted metabolomics approach using tandem mass spectrometry (LC/MS-MS), one can quantify the absolute concentration of metabolites in a biological sample in a high throughput fashion. We seek to answer how altered tryptophan metabolism may affect the production of gut microbiome-derived metabolites in mice. At the completion of this work, no known complete metagenome model exists that accounts for all reactions present in all bacterial species present in the murine gut, so we could not identify the ShReD-based module that incorporates tryptophan metabolism. As an alternative, we utilize a probabilistic search algorithm to identify a set of metabolites, whose concentrations would be directly influenced by a change in tryptophan metabolism. We then quantify those metabolite concentrations in cecum and feces samples of mice for two different age groups.

2 Identification of Biochemical Network Modules Based on Shortest Retroactive Distances

2.1 Abstract

Modularity analysis offers a route to better understand the organization of cellular biochemical networks as well as to derive practically useful, simplified models of these complex systems. While there is general agreement regarding the qualitative properties of a biochemical module, there is no clear consensus on the quantitative criteria that may be used to systematically derive these modules. In this work, we investigate cyclical interactions as the defining characteristic of a biochemical module. We utilize a round trip distance metric, termed Shortest Retroactive Distance (ShReD), to characterize the retroactive connectivity between any two reactions in a biochemical network and to group together network components that mutually influence each other. We evaluate the metric on two types of networks that feature feedback interactions: (i) epidermal growth factor receptor (EGFR) signaling and (ii) liver metabolism supporting drug transformation. For both networks, the ShReD partitions found hierarchically arranged modules that confirm biological intuition. In addition, the partitions also revealed modules that are less intuitive. In particular, ShReD-based partition of the metabolic network identified a 'redox' module that couples reactions of glucose, pyruvate, lipid and drug metabolism through shared production and consumption of NADPH. Our results suggest that retroactive interactions arising from feedback loops and metabolic cycles significantly contribute to the modularity of biochemical networks. For metabolic networks, cofactors play an important role as allosteric effectors that mediate the retroactive interactions.

2.2 Introduction

Hierarchical modularity has emerged as an organizational principle of biochemical networks, where larger less cohesive clusters of network components (e.g. metabolic enzymes or signaling molecules) comprise functionally distinct sub-clusters (Ravasz *et al.*, 2002; Papin *et al.*, 2004). For example, Ihmels and coworkers analyzed the co-expression patterns of metabolic genes in *Saccharomyces cerevisiae* to find coordinated regulation of individual pathways as well as higher-order functions such as biosynthesis and stress response that require multiple feeder pathways (Ihmels *et al.*, 2004). Hierarchical organization was also observed by Gutteridge and coworkers for metabolic regulatory networks, where hub metabolites regulating many enzymes connect to modules of spoke metabolites that are chemically similar and/or regulate functionally related enzymes (Gutteridge *et al.*, 2007).

In recent years, observations on modularity have prompted metabolic engineers and synthetic biologists to consider whole pathways, rather than individual genes, as modular building units for cellular design (Heinemann and Panke, 2006). An emerging design rule is to assemble and express a coherent set of genes that encode the desired biochemical pathway along with regulatory mechanisms that modulate the activity of the pathway (Andrianantoandro *et al.*, 2006). Modularity analysis also offers a route to build practically useful, simplified models of complex biological systems. The size and complexity of biochemical networks reconstructed from genome databases has greatly increased over the years (Minoru Kanehisa *et al.*, 2006, 2010; Ogata *et al.*, 1999), rendering the estimation of kinetic or regulatory parameters either impractical or outright infeasible. In this regard, the modularity of a biochemical network should allow the system to be partitioned into minimally interdependent parts, enabling systematic derivation of coarse-grained, yet comprehensive models. Such coarse-grained models could greatly simplify the parameter estimation problem by substituting detailed reaction kinetics with less detailed module kinetics (Riel and Sontag).

While there is general agreement that a biochemical module should represent a group of connected network components, and that the arrangement of modules in the network is hierarchical, there is less consensus on the criteria that should be used to systematically extract biologically meaningful modules (C. L. Barrett *et al.*, 2009; Ederer *et al.*, 2003; S Schuster *et al.*, 2002; J. Zhao *et al.*, 2006). One recent argument was to focus on cyclical, or 'retroactive,' interactions between network components, as opposed to simple connectivity (Saez-Rodriguez *et al.*, 2008). Biochemical pathways operate with direction, where upstream components (e.g. concentration of reactants) influence downstream components (e.g. concentration of products). In the case where a downstream component also influences an upstream component (e.g. via a feedback regulatory mechanism), the two components participate in a cycle and thus interact retroactively. Placing such components into the same module reduces the interdependence between different modules, consistent with the intuitive definition of a biological module. Indeed, metabolic cycles and feedback loops have been shown to confer robustness (Kitano, 2004) by isolating external perturbations and attenuating their propagation through the entire network (Stelling *et al.*, 2004).

In this chapter, we extend the concept of retroactivity to account for cyclical interactions spanning distant parts of a biochemical network as exemplified by feedback loops of signaling and metabolic pathways. In earlier work (Saez-Rodriguez *et al.*, 2008), retroactivity was only considered for interactions between nearest neighbors in a network. To investigate hierarchy, we adopted Newman's algorithm for community detection (M. E. J. Newman, 2006) to successively partition a network into modules containing cyclical interactions based on a round trip distance metric, which we call Shortest Retroactive Distance (ShReD). Applied to test models of a signaling network (Oda *et al.*, 2005) (Figure 2-1) and a metabolic network (Figure 2-2), the ShReD-based partitions produced hierarchically arranged modules that confirm biological knowledge.

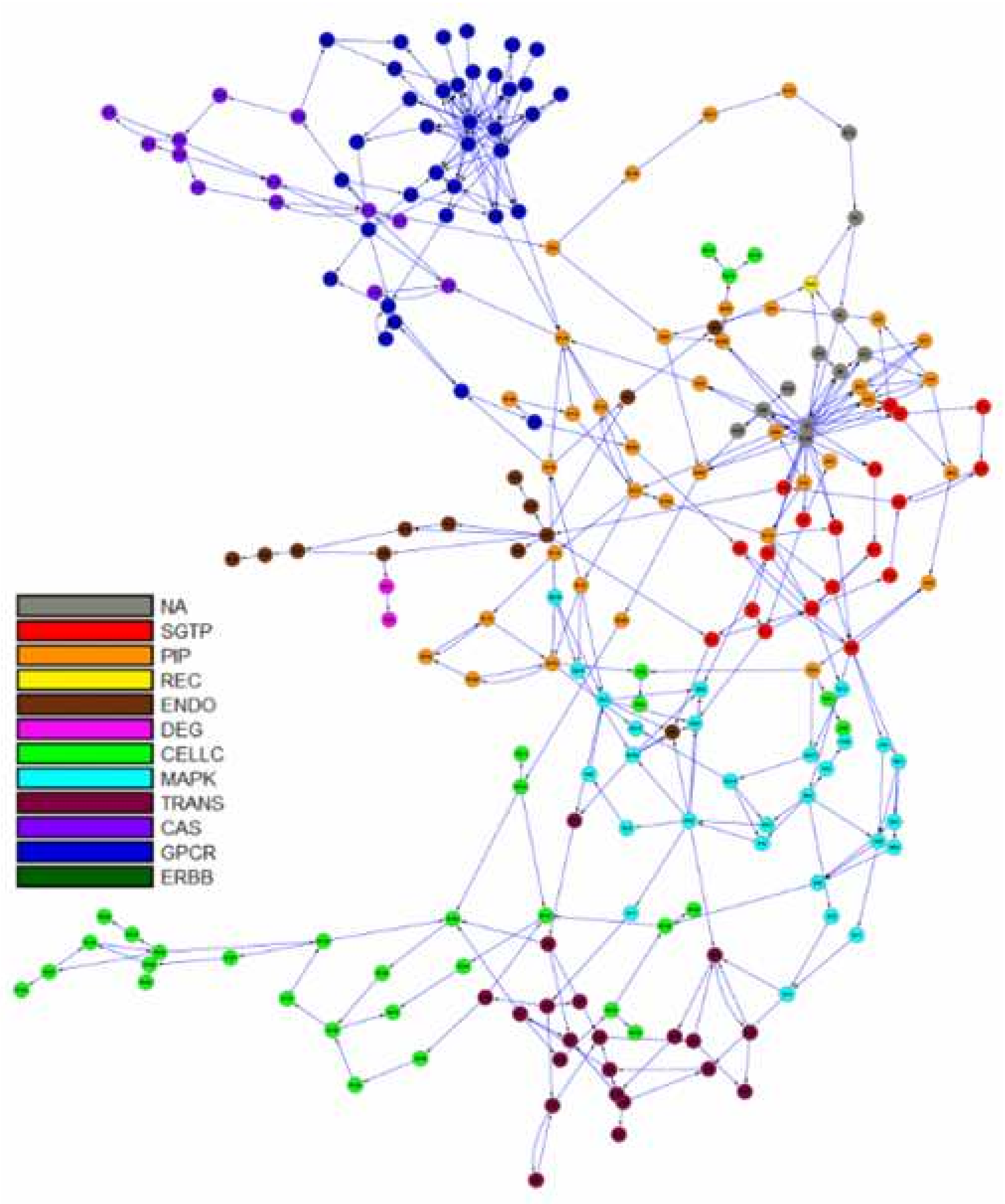


Figure 2-1 Graph image of the signaling network

Each reaction in the network was *a priori* assigned to one of 11 canonical signaling pathways as described in Methods. The pathway assignments are indicated by the color of the reaction vertex in the network. (NA: not assigned; SGTP: small guanosine triphosphatase mediated signaling; PIP: phosphatidylinositol polyphosphate signaling; REC: recycling; ENDO: endocytosis; DEG: degradation; CELLC: cell cycle; MAPK: mitogen-activated protein kinase cascade; TRANS: transcription; CAS: Ca^{2+} signaling; GPCR: G-protein coupled receptor mediated signaling; ERBB: erythroblastic leukemia viral oncogene homolog receptor signaling)

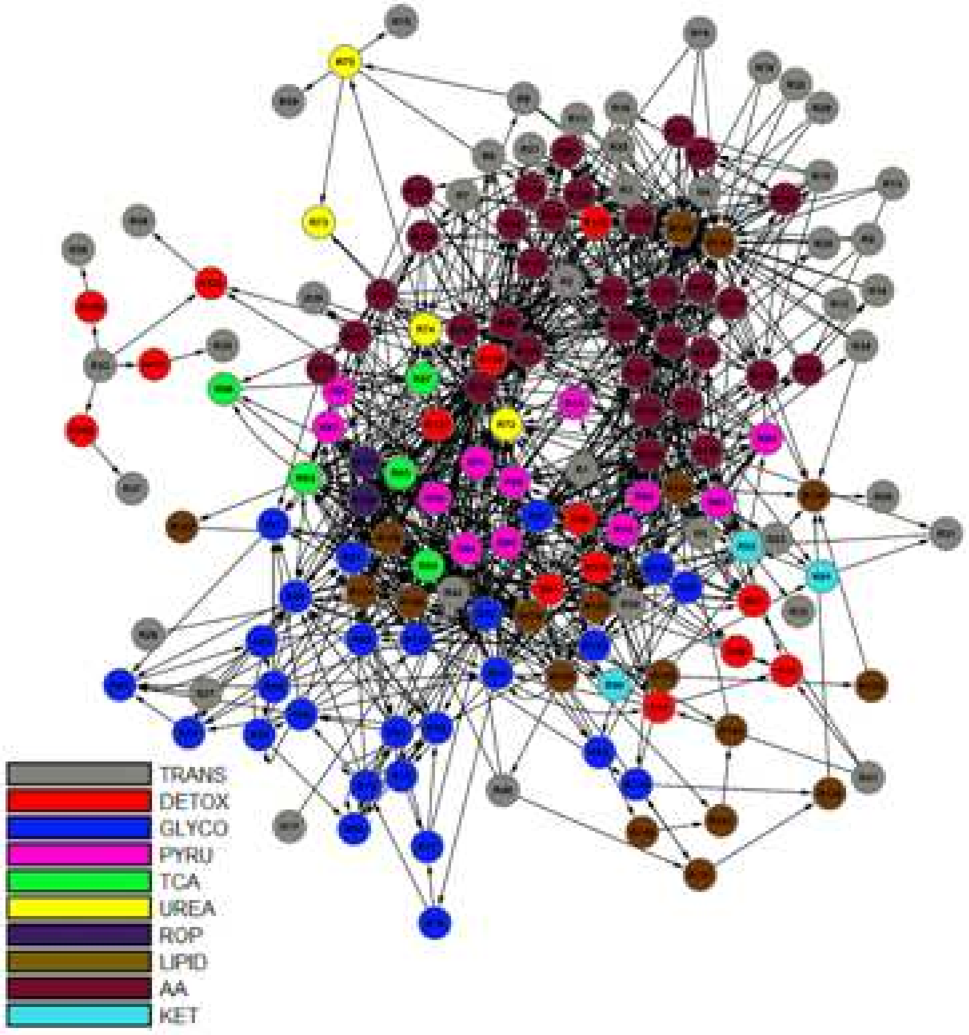


Figure 2-2: Graph image of hepatocyte metabolic network

Each reaction in the network was *a priori* assigned to one of 10 textbook pathways as described in Methods. The pathway assignments are indicated by the color of the reaction vertex in the network. (TRANS: transport; DETOX: detoxification; GLYCO: glucose metabolism; PYRU: pyruvate metabolism; TCA: tca cycle; UREA: urea cycle; ROP: oxidative phosphorylation; LIPID: lipid metabolism; AA: amino acid metabolism; KET: ketone body metabolism)

In addition, the partitions also revealed modules that are less intuitive. For the metabolic network, we also examined the role of allosteric regulators and cofactors as network elements that determine the number of cyclical interactions and the hierarchical depth of modules.

2.3 Methods

2.3.1 Network Representation

A common way to model a biochemical network using a graph is to represent the components as vertices and their interactions as edges. In this study, the focus is on understanding the hierarchical and modular relationship among reactions, treating metabolites as shared resources among modules. We therefore use a directed graph with vertices representing reactions and edges indicating a directional interaction between the connected reactions. Edges are drawn between two reactions (Figure 2-3a) if the product of one reaction is either a reactant (Figure 2-3b) or allosteric effector of another reaction (Figure 2-3c). For reversible reactions, reactant-product relationships are considered in both directions.

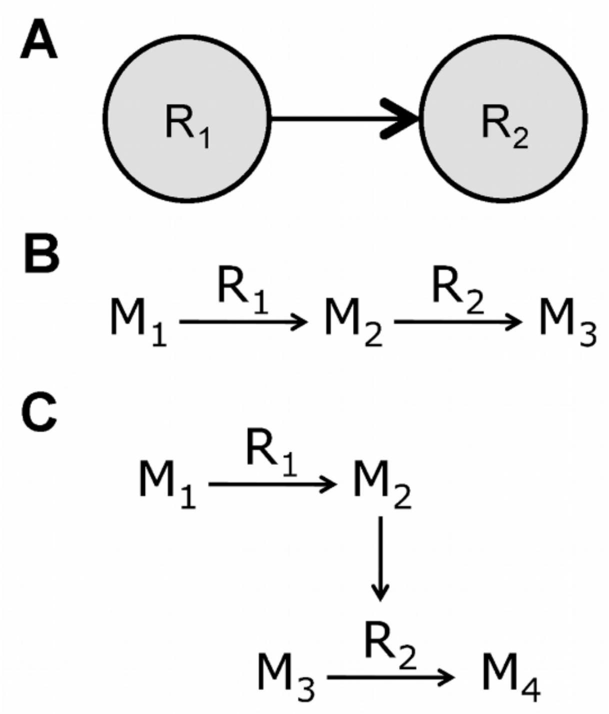


Figure 2-3: Network Representation

(A) A reaction-centric representation of two different cases (B and C) where one reaction is upstream of another. (B) Reaction R_1 produces a metabolite M_2 that is consumed by reaction R_2. (C) Reaction R_1 produces a metabolite M_2 that is an allosteric effector of the enzyme catalyzing reaction R_2.

2.3.2 Shortest Retroactive Distance

We utilize round trip distance as a metric, which we call Shortest Retroactive Distance (ShReD), to characterize the connectivity between two vertices that interact retroactively. A retroactive interaction exists between two vertices *i* and *j*, if and only if there is a directional path from vertex *i* to *j and* a return path from vertex *j* to *i*. The retroactive interaction represents a mechanism for mutual feedback, and thus expresses interdependence. The ShReD of vertices *i* and *j* ($ShReD_{ij}$) is the sum of the *shortest path* distance from node *i* to *j* and the shortest return path distance from node *j* to *i*. In the example network of Figure 2-4, $ShReD_{1,3}$ is 3 because there are two edges along the shortest path from R_1 to R_3 and there is one edge from R_3 to R_1.

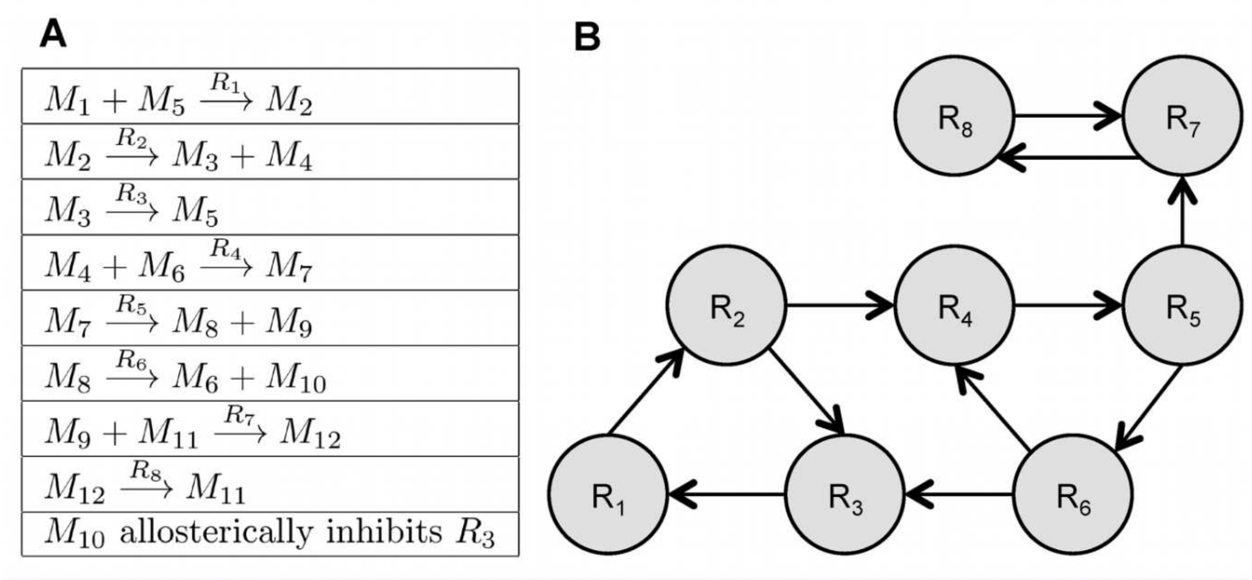

A
$M_1 + M_5 \xrightarrow{R_1} M_2$
$M_2 \xrightarrow{R_2} M_3 + M_4$
$M_3 \xrightarrow{R_3} M_5$
$M_4 + M_6 \xrightarrow{R_4} M_7$
$M_7 \xrightarrow{R_5} M_8 + M_9$
$M_8 \xrightarrow{R_6} M_6 + M_{10}$
$M_9 + M_{11} \xrightarrow{R_7} M_{12}$
$M_{12} \xrightarrow{R_8} M_{11}$
M_{10} allosterically inhibits R_3
B
R_8
R_7
R_2
R_4
R_5
R_1
R_3
R_6

Figure 2-4: Example Illustrating ShReD-based Network Partition

(A) The example network comprises 8 reactions and 1 allosteric inhibition. (B) Graph representation of the reaction-to-reaction interactions in the example network.

There is another cycle connecting the two reaction vertices, which also involves R_4, R_5 and R_6. This cycle, however, is not the ShReD, as its length of 6 exceeds the ShReD value of 3. For a given network (or sub-network) a ShReD value is computed for every pair of vertices in the network (or sub-network). To compute the ShReD values, we first calculated the shortest distances between all pairs of vertices using the Floyd-Warshall algorithm (Floyd, 1962). The resulting all-pairs shortest path matrix was then added to its own transpose to generate a symmetrical ShReD matrix. When there is no path or no return path between two vertices, the ShReD value between these two vertices is infinity. The ShReD between a node and itself is zero. For the example network in Figure 2-4, the ShReD matrix is as follows:

$$\textbf{ShReD} = \left[\begin{array}{ccccccccc} & R_1 & R_2 & R_3 & R_4 & R_5 & R_6 & R_7 & R_8 \\ R_1 & 0 & 3 & 3 & 6 & 6 & 6 & \infty & \infty \\ R_2 & 3 & 0 & 3 & 6 & 6 & 6 & \infty & \infty \\ R_3 & 3 & 3 & 0 & 6 & 6 & 6 & \infty & \infty \\ R_4 & 6 & 6 & 6 & 0 & 3 & 3 & \infty & \infty \\ R_5 & 6 & 6 & 6 & 3 & 0 & 3 & \infty & \infty \\ R_6 & 6 & 6 & 6 & 3 & 3 & 0 & \infty & \infty \\ R_7 & \infty & \infty & \infty & \infty & \infty & \infty & 0 & 2 \\ R_8 & \infty & \infty & \infty & \infty & \infty & \infty & 2 & 0 \end{array}\right] \tag{2.1}$$

2.3.3 Partition Algorithm

Partitions were obtained by adapting Newman’s community detection algorithm (M. E. J. Newman, 2006), which was modified to generate partitions based on the ShReD metric, as opposed to simple connectivity. An overview of the algorithm flow is shown in Figure 2-5.

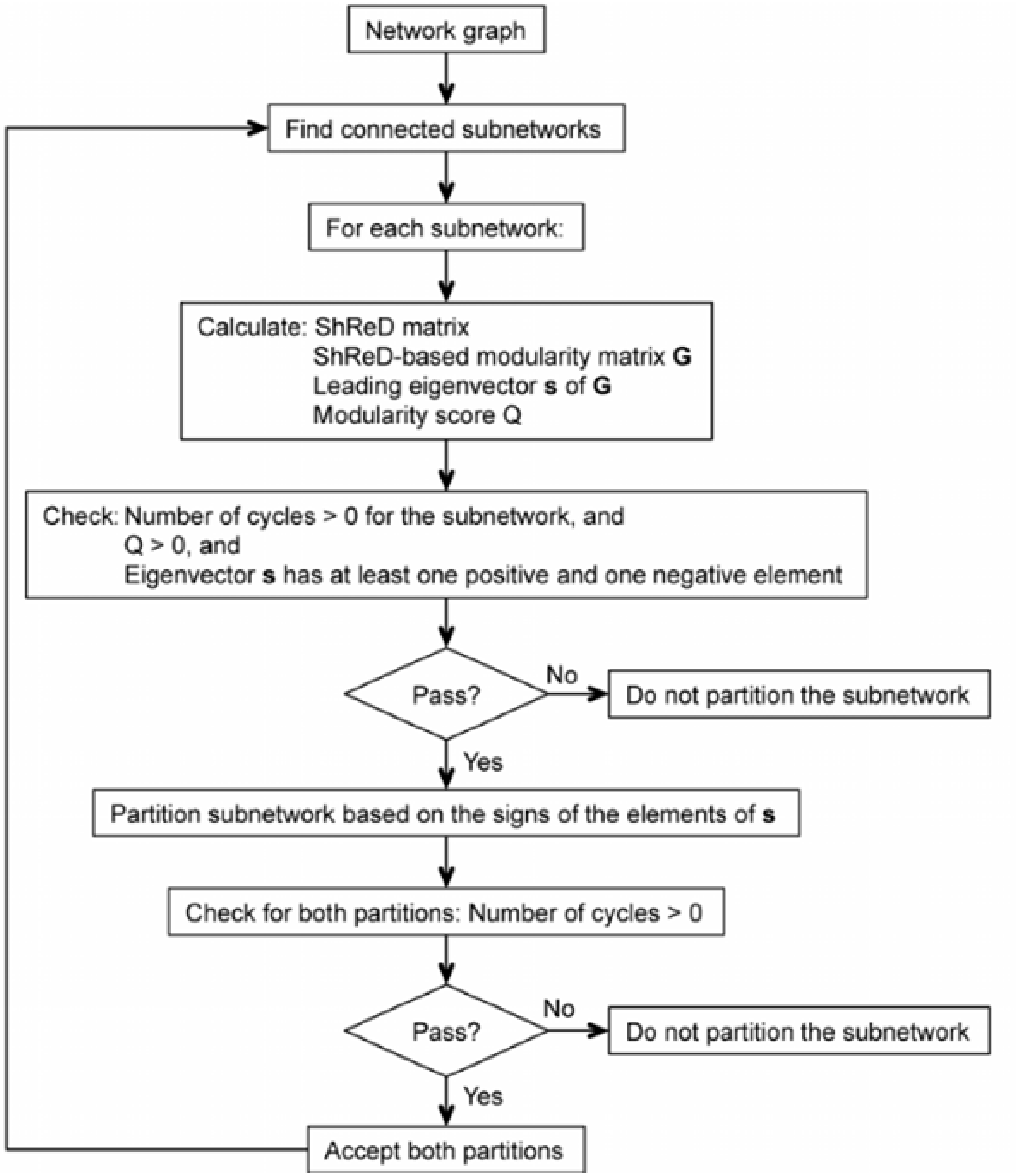

Network graph
Find connected subnetworks
For each subnetwork:
Calculate: ShReD matrix
ShReD-based modularity matrix **G**
Leading eigenvector **s** of **G**
Modularity score Q
Check: Number of cycles > 0 for the subnetwork, and
Q > 0, and
Eigenvector **s** has at least one positive and one negative element
Pass?
No
Do not partition the subnetwork
Yes
Partition subnetwork based on the signs of the elements of **s**
Check for both partitions: Number of cycles > 0
Pass?
No
Do not partition the subnetwork
Yes
Accept both partitions

Figure 2-5:Schematic illustrating the flow of the partition algorithm

The initial step is to find the connected subnetworks in the parent network using a breadth-first traversal algorithm (TH, 2009), as it is possible that the parent network, represented as a reaction centric graph, may not be connected. For the search, the network is represented as an undirected graph, as we are interested in identifying the connectivity of vertices, regardless of direction. Each connected subnetwork is then partitioned into two daughter subnetworks to maximize a "modularity score" while ensuring that each subnetwork resulting from a partition retains at least one retroactive interaction, i.e. cycle. Applied recursively, the algorithm produces a hierarchical tree of binary partitions.

In Newman's algorithm, the modularity score was computed as the difference between the actual and expected number of *connections* between two components. In this study, we computed the difference between the actual and expected ShReD to determine the modularity score. The expected ShReD between i and j, P_{ij}, is computed as the arithmetic mean of the average of all non-zero and non-infinite ShReDs involving i and the average of all non-zero and non-infinite ShReDs involving j:

$$P_{ij} = \frac{1}{2}\left[\frac{\sum_{k=1}^{n} ShReD_{ik}}{D_i} + \frac{\sum_{k=1}^{n} ShReD_{jk}}{D_j}\right] \text{ for } \begin{cases} ShReD_{ik} \neq 0, \infty \\ ShReD_{jk} \neq 0, \infty \end{cases} \tag{2.2}$$

where D_i and D_j are the number of non-zero and non-infinite ShReDs involving i and j respectively, and n is the total number of vertices in the network (or sub-network). We define a *ShReD-based modularity matrix*, **G**, as follows:

$$G_{ij} = P_{ij} - ShReD_{ij} \tag{2.3}$$

The diagonal entries of **G** are set to zero, because both the expected and actual ShReD between a vertex and itself are zero. An entry G_{ij} is also set to zero, if $ShReD_{ij}$ is infinity. For the example network in Figure 2-4, the average ShReD of R_1 and R_2 are both 4.8. The expected ShReD between R$_1$ and R$_2$, P_{12}, is thus 4.8, and G_{12} is 1.8. The full matrix **G** for the example network is shown below. The ShReD-based modularity matrix differs from Newman's *connectivity-based modularity matrix*, which does not take into account the direction of an interaction.

$$\mathbf{G} = \left[\begin{array}{c|cccccccc} & R_1 & R_2 & R_3 & R_4 & R_5 & R_6 & R_7 & R_8 \\ \hline R_1 & 0 & 1.8 & 1.8 & -1.2 & -1.2 & -1.2 & 0 & 0 \\ R_2 & 1.8 & 0 & 1.8 & -1.2 & -1.2 & -1.2 & 0 & 0 \\ R_3 & 1.8 & 1.8 & 0 & -1.2 & -1.2 & -1.2 & 0 & 0 \\ R_4 & -1.2 & -1.2 & -1.2 & 0 & 1.8 & 1.8 & 0 & 0 \\ R_5 & -1.2 & -1.2 & -1.2 & 1.8 & 0 & 1.8 & 0 & 0 \\ R_6 & -1.2 & -1.2 & -1.2 & 1.8 & 1.8 & 0 & 0 & 0 \\ R_7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ R_8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \tag{2.4}$$

Defining the modularity score Q based on the ShReD-based modularity matrix **G**, we wish to find a vector **s**, which assigns each vertex in the network to one of the two partitioned sub-networks to maximize *Q:*

$$Q = \sum_i \sum_j G_{ij} s_i s_j \tag{2.5}$$

where s_i is an element of a vector **s.** Each s_i has a value of either -1 or 1. An increase in Q is obtained in two cases: if G_{ij} is positive and the vertices i and j are assigned to the same sub-network (s_i = s_j = 1 or s_i = s_j = -1), or if G_{ij} is negative and the two vertices are assigned to different subnetworks (s_i = 1 and s_j = -1 or vice versa). The vector **s** maximizing Q can be found using spectral partitioning methods (Pothenf *et al.*, 1990) as described by Newman(M. E. J. Newman, 2006). The solution to the maximization problem can be approximated by the leading eigenvector of **G**. For our example network (Figure 12), the leading eigenvector of **G** (Equation 2.4) is given by **v = [**-0.41, -0.41, -0.41, 0.41, 0.41, 0.41, 0, 0], from which **s** is approximated as **s** = [-1, -1, -1, 1, 1,1, -1, -1]. All non-positive entries, including zero, in the eigenvector are assigned the value -1. This partition assigns R_1, R_2, R_3, R_7 and R_8 to one module, and R_4, R_5 and R_6 to the other module. The reactions in the first module are not fully connected, which gives rise to two disconnected components, one comprising R_1, R_2 and R_3 and the other comprising R_7 and R_8. In this example, a single binary partition generated three separate modules, each consisting of a single cycle.

In Newman's original community detection algorithm, partitioning of a subnetwork continues if the modularity score Q is greater than zero and the leading eigenvector $\mathbf{s}$ of the modularity matrix $\mathbf{G}$ has at least one positive and one negative element; otherwise the subnetwork is not further partitioned. The algorithm terminates if there is no subnetwork that can be further partitioned. In our algorithm, we modified the termination criterion to also check that there is a cycle in each subnetwork resulting from a partition operation. The check for a cycle was performed using an algorithm similar to topological sort (A. B. Kahn, 1962). For a given module abstracted as a directed graph, the number of incoming edges is computed for each vertex. A vertex with zero incoming edges is removed from the graph along with its outgoing edges. The number of incoming edges is then recalculated for the remaining vertices. The process repeats until there are no more vertices, in which case the graph has no cycles, or until there are no vertices with zero incoming edges, indicating the presence of a cycle. In our example, the Q score for the first partition is greater than zero (Q = 43.2) and each resulting subnetwork contained at least one cycle. Thus, the partitioned subnetworks are accepted as modules and the algorithm continues by finding the connected subnetworks in each module. The module comprising R_1, R_2, R_3, R_7 and R_8 is not fully connected, and two subnetworks are found, one comprising R_1, R_2, and R_3 and the other comprising R_7 and R_8. Neither subnetwork can be further partitioned, as every element in the leading eigenvector of the corresponding modularity matrix has the same sign. Similarly, the module comprising R_4, R_5 and R_6 cannot be further partitioned, as every element in the leading eigenvector of the corresponding modularity matrix has the same sign, and the algorithm terminates.

2.3.4 Hierarchical Tree of Partitions

The partitioning results are reported in the form of a hierarchical tree annotated with several properties. Each module is represented as a pie chart, where the size of each slice is proportional to the fraction of reactions that belong to the corresponding, pre-assigned canonical (textbook) grouping. The *homogeneity index* of a module corresponds to the fraction occupied by the largest slice in the pie chart. The homogeneity index therefore ranges from 0 to 1, where a larger number indicates greater homogeneity in terms of composition based on the canonical group assignments. The black lines connecting the nodes in the hierarchical tree represent ShReD-based partitions, whereas the red lines represent the formation of components from partitions that include disconnected components. The depth of a module is determined as the number of black edges traversed from the root node to the module. The height of a module is determined as the largest possible number of black edges traversed from the module to a terminal leaf node.

The *number of cycles* within a module is used to compare the partitions obtained based on the ShReD and Newman's connectivity metrics. While standard algorithms exist for counting the number of cycles in a graph (Johnson, 1977), the run time is proportional to the number of (non-unique) cycles. The number of cycles may be exponential in the number of vertices, and renders cycle counting as computationally inefficient. The cycle count is thus reported up to 1,000 unique cycles. Any count above 1,000 is effectively reported as 1,000. In addition to cycles, we also determined the number of *non-infinite* shortest retroactive paths in a module as well as the mean ShReD of the module. The mean ShReD of a module is calculated by averaging the corresponding non-infinite entries in the corresponding ShReD matrix.

2.3.5 Models

As case studies, we examined two types of biochemical networks that feature directed interactions and feedback loops.

2.3.5.1 Signaling Model

The signaling network was reconstructed based on a published model of epidermal growth factor receptor (EGFR) signaling (Oda *et al.*, 2005). The model was downloaded as an SBML file and cast into the form of a stoichiometric matrix based on the directional interactions between signaling molecules defined in the SBML file. The model consisted of 322 signaling molecules (metabolites and proteins) participating in 211 signaling reactions. In addition to the signaling reactions, the model includes 238 allosteric interactions between the signaling molecules and reactions. The reactions in this model were *a priori* assigned to groups based on their previously catalogued function (Oda *et al.*, 2005). For example, the reactions that convert ERK1/2 and PKB/akt into their active forms were assigned to the MAPK cascade and PIP signaling, respectively. This initial grouping, which reflects historical knowledge of signaling modularity, provided a basis for comparison between biological knowledge-driven, canonical associations versus partition-driven, systematically obtained network modules. The graph model for this network was shown in Figure 2-1.

2.3.5.2 Metabolic Model

A stoichiometric network model of human hepatocyte metabolism was reconstructed from the KEGG reaction database and further augmented by the addition of xenobiotic transformation reactions, as well as regulatory interactions mediated by allosteric effectors. The model comprised 159 reactions, 146 metabolites, and 61 regulatory interactions. The xenobiotic transformation reactions were added to describe the metabolism of the anti-diabetic compound troglitazone (TGZ), including steps needed to supply conjugation substrates such as glutathione (GSH). The regulatory interactions in the model reflect known allosteric effects of metabolites on reaction activity as described in a standard biochemistry textbook (Nelson and Cox, 2008). Information about the allosteric effects of metabolites was organized into a regulatory matrix, where the columns and rows represented the effector metabolites and reactions regulated by these metabolites, respectively. The inhibition or activation of a reaction *j* by an allosteric effector *i* was denoted by a negative one (-1) or positive one (+1), respectively, in the matrix element (*i*, *j*). For all other cases where there were no known allosteric interactions, a zero (0) was entered into the corresponding matrix element. For example, the TCA cycle intermediate citrate allosterically inhibits phosphofructokinase I (PFK) to regulate the flux of glucose-derived substrates into the TCA cycle. In the regulatory matrix, this feedback inhibition is denoted by the value (-1) in the matrix element corresponding to (*citrate*, *PFK*). Similar to the EGFR signaling model, the reactions of the metabolic model were assigned into one of the following groups based on their canonical memberships as defined in standard biochemistry textbooks: transport (TRANS), detoxification (DETOX), sugar metabolism (encompassing glycolysis, gluconeogenesis, pentose phosphate shunt, and glycogen metabolism) (GLYCO), pyruvate metabolism (PYRU), TCA cycle (TCA), urea cycle (UREA), oxidative phosphorylation (ROP), lipid metabolism (LIPID), amino acid synthesis and degradation (AA), and ketone body production (KET). To test the impact of allosteric regulation on modularity, separate graph models were constructed by either omitting regulatory edges altogether, or just removing the cofactors (i.e. ATP, NADH and NADPH), which represented the majority of allosteric effectors. The cofactors were removed from the stoichiometric network model by deleting the corresponding rows of the stoichiometric matrix, which eliminated the regulatory edges reflecting cofactor-driven dependencies between reactions.

2.4 Results

2.4.1 Effect of Retroactive Interactions on Modularity: Signaling Network

To examine the effect of cyclical, i.e. retroactive, interactions on modularity, we compared the partitions of the EGFR signaling network obtained using Newman's connectivity (Figure 2-6a) and the ShReD metric (Figure 2-6b).

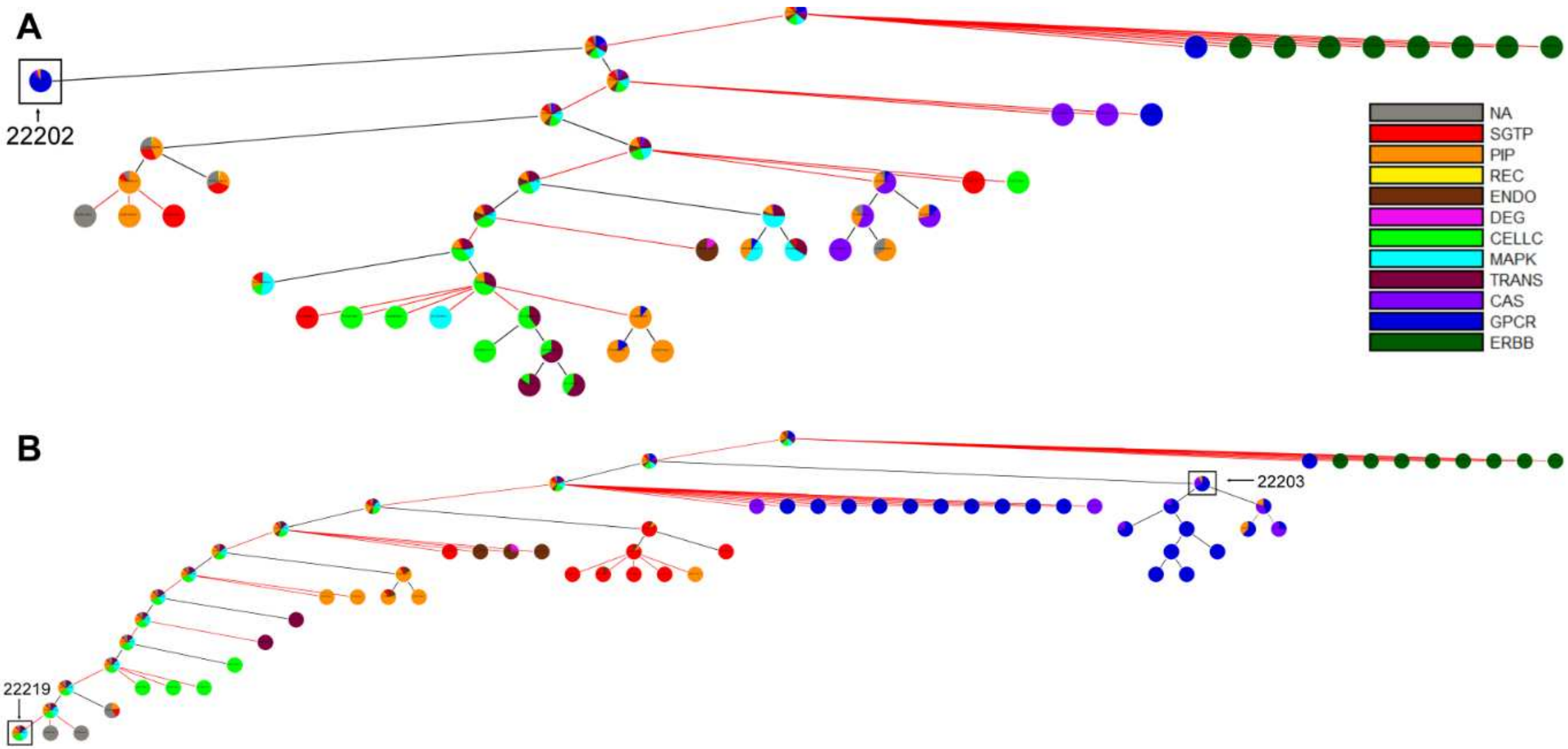


Figure 2-6: Hierarchical partitions of EGFR signaling network

(A) Partitions obtained using Newman's connectivity metric. The GPCR dominated module (ID: 22202) has 36 reactions and 28 cycles. (B) Partitions obtained using the ShReD metric. The GPCR dominated module (ID: 22203) has 39 reactions and 167 cycles. The terminal node (ID: 22219) has 99 reactions, but only 10 cycles.

Several qualitative similarities between the two partitions are evident. In both partitions, modules that possess a large fraction of reactions from phosphatidylinositol polyphosphate (PIP) signaling coupled to either intracellular Ca^{2+} signaling (CAS) or small guanosine triphosphatase (SGTP) were identified. Quantitatively, both partitions reach a hierarchical depth of 6 and become more homogeneous closer to the terminal nodes of the partition tree. From the root to terminal nodes, the canonical group compositions of the modules (represented by the pie colors) trend toward a single, dominant group (Figure 2-7). At the terminal nodes (height zero), the fraction of reactions in a module belonging to a single canonical group, on average, exceeds 80 % for both Newman and ShReD partitions.

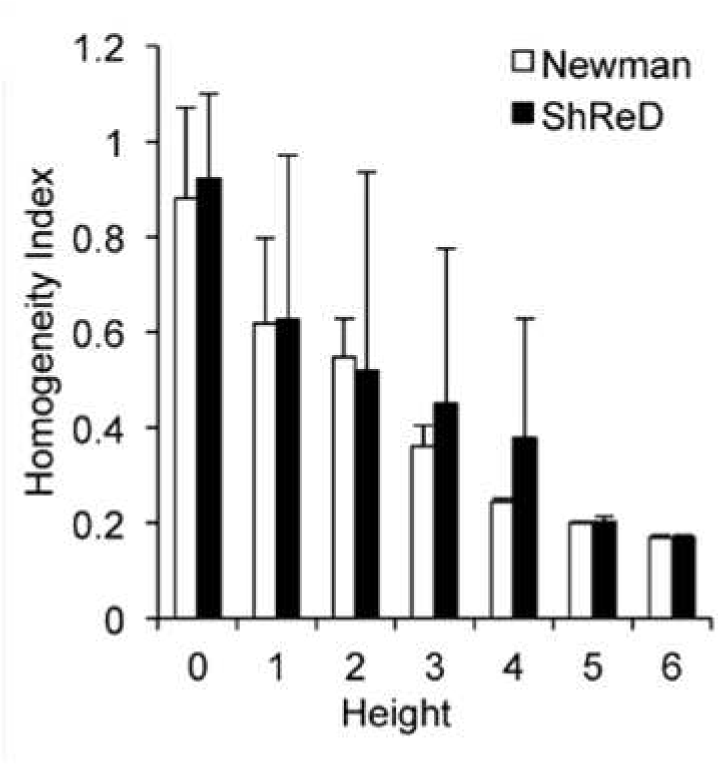


Figure 2-7: Homogeneity of modules as a function of partition height

See methods for definition of homogeneity index

There are also notable differences between the two partitions. While both partitions extract modules predominantly consisting of G-Protein coupled Receptor (GPCR) activation reactions, the ShReD partition identifies greater hierarchy stemming from those modules. In the Newman partition, there are several terminal leaf nodes that predominantly comprise Mitogen Activated Protein Kinase (MAPK) reactions. Analogous terminal nodes are not present in the ShReD partition. The ShReD partition yields a large terminal node consisting of 99 reactions (Figure 2-8, ID: 22219), whereas the largest terminal node of the Newman partition consists of 36 reactions (Figure 2-9, ID: 22202).

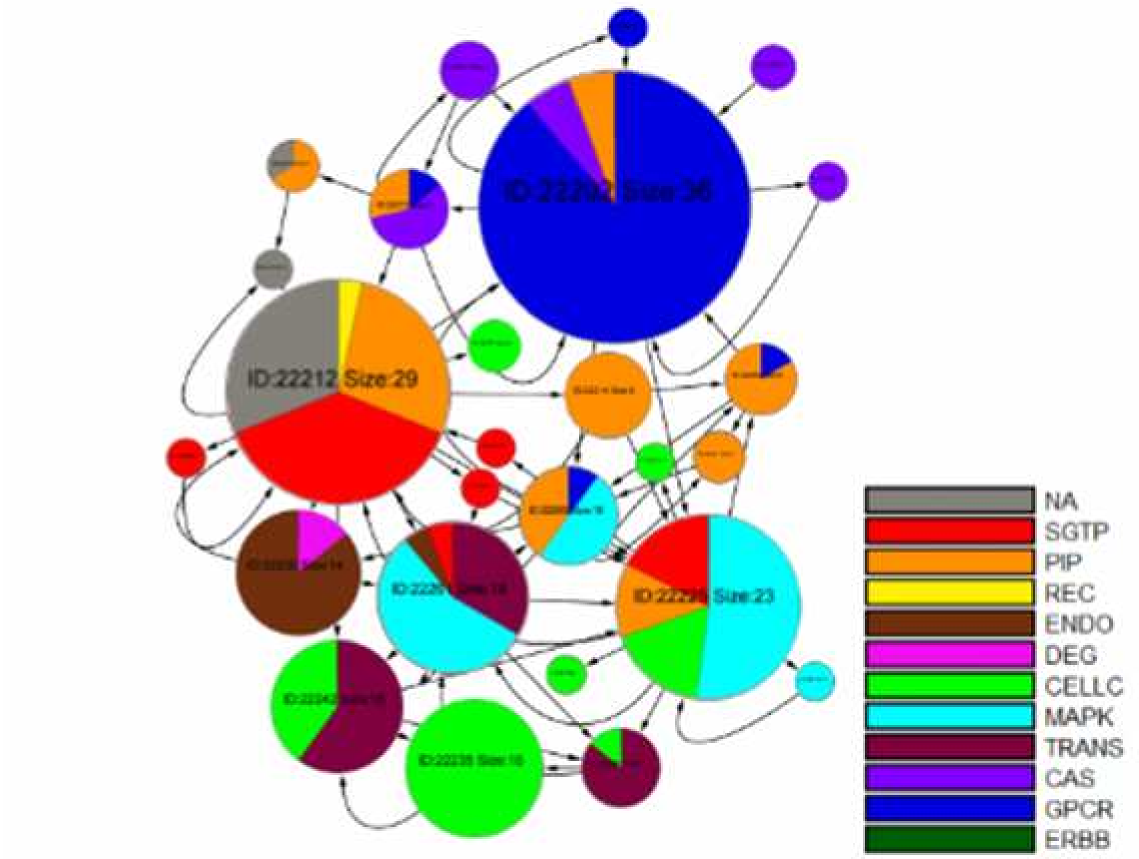


Figure 2-8: Network of terminal modules for EGFR model: Newman Partition

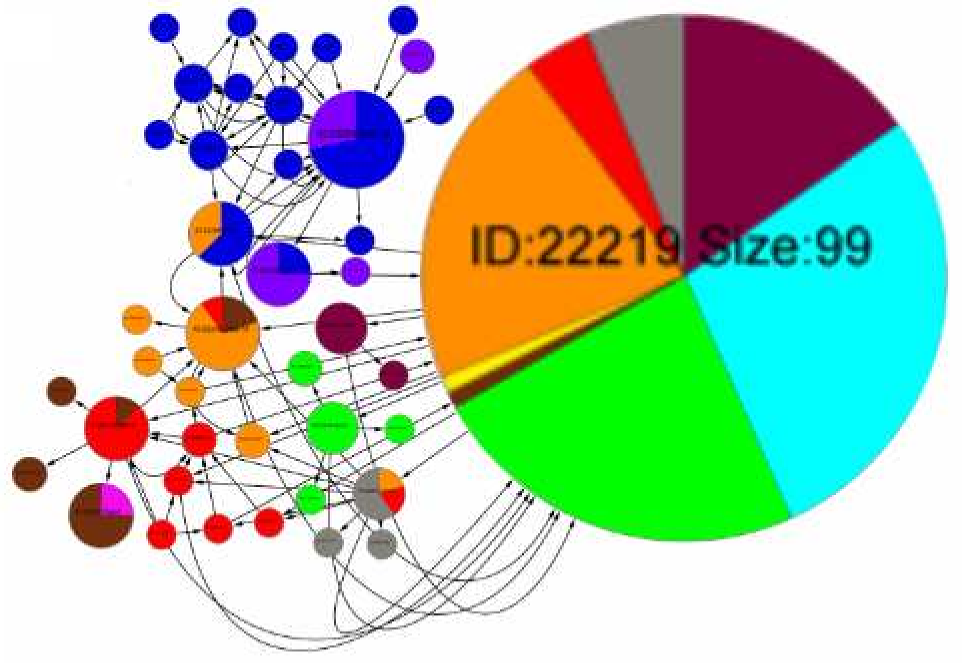


Figure 2-9: Network of terminal modules: ShReD-based partition.

The largest terminal node in the Newman partition (Figure 2-8, ID: 22202) predominantly comprises GPCR transactivation reactions, whereas the largest terminal node in the ShReD partition (Figure 2-9, ID: 22219) comprises several signaling functions, including MAPK cascade, endocytosis, and cell cycle. Another notable difference is that while the average number of cycles in a module decreases with increasing depth for both partitions, a larger number of cycles are preserved in the ShReD partition at greater depths (Figure 2-10).

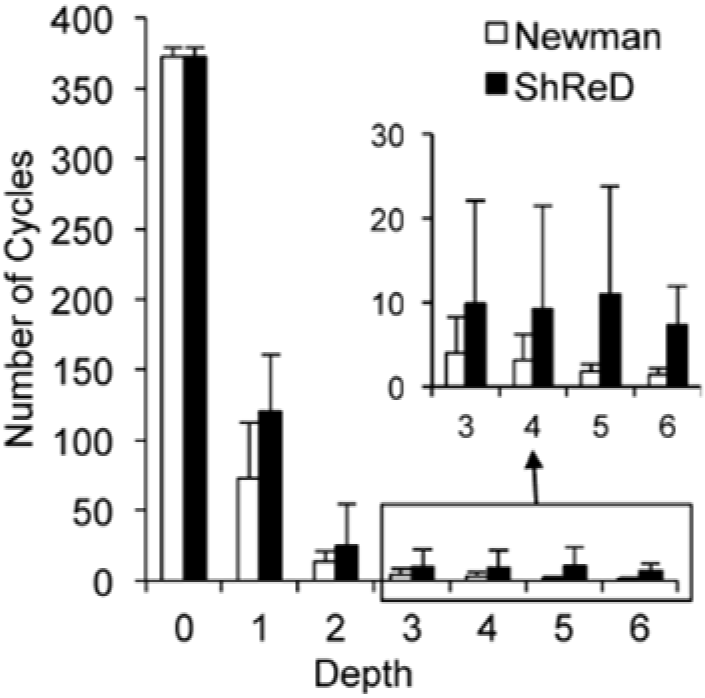


Figure 2-10: Number of Cycles as a function of Depth for EGFR model partitions

2.4.2 Effect of Retroactive Interactions on Modularity:

We next compared the Newman (Figure 2-11a) and ShReD partitions (Figure 2-11b) for the liver metabolic network, complete with regulatory edges and cofactors.

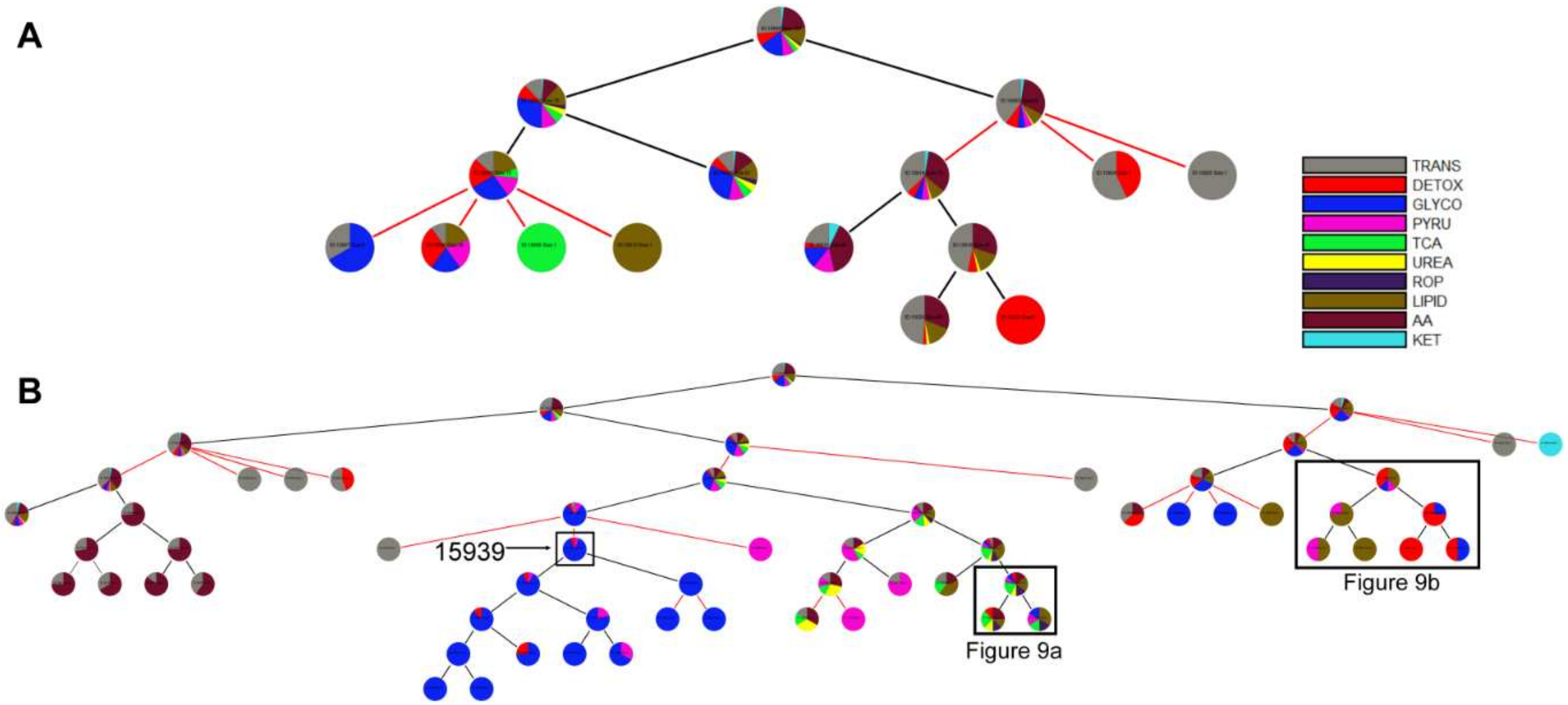


Figure 2-11: Hierarchical Partitions of Hepatocyte Metabolic Network

(A) Partitions obtained using Newman's connectivity metric. (B) Partitions obtained using the ShReD. Details of the reactions in the boxed modules are shown in Figures 9a and 9b. The other boxed module (ID: 15939) contains pyruvate kinase.

As was the case for the EGFR network, both partitions lead to modules that generally increase in homogeneity from the root node to the terminal nodes (Figure 2-12).

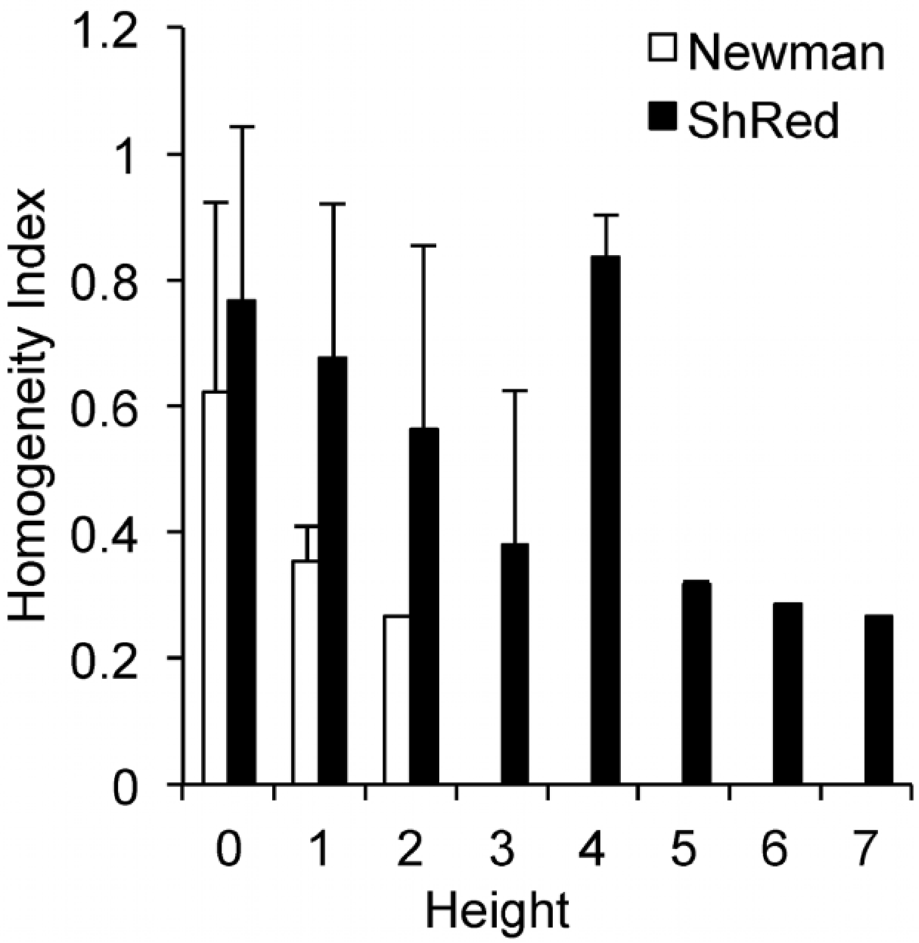


Figure 2-12: Effect of the partition metric on the homogeneity index of hepatocyte network modules

The height of the root node in the Newman partition tree is 2, whereas the height of the ShReD tree is 7. Error bars represent one standard deviation.

However, unlike the EGFR network, the arrangement and compositions of the two partitions are drastically different (Figure 2-11). In contrast to the Newman partition, the ShReD partition generates modules with hierarchical depth, similar to the GPCR dominated modules of the EGFR network. In the case of the metabolic network, hierarchical depth was greatest for modules comprising reactions in and around glycolysis (GLYCO). Moreover, the terminal node modules of the ShReD partition reach greater homogeneity compared to the Newman partition (Figure 2-12, Figure 2-13).

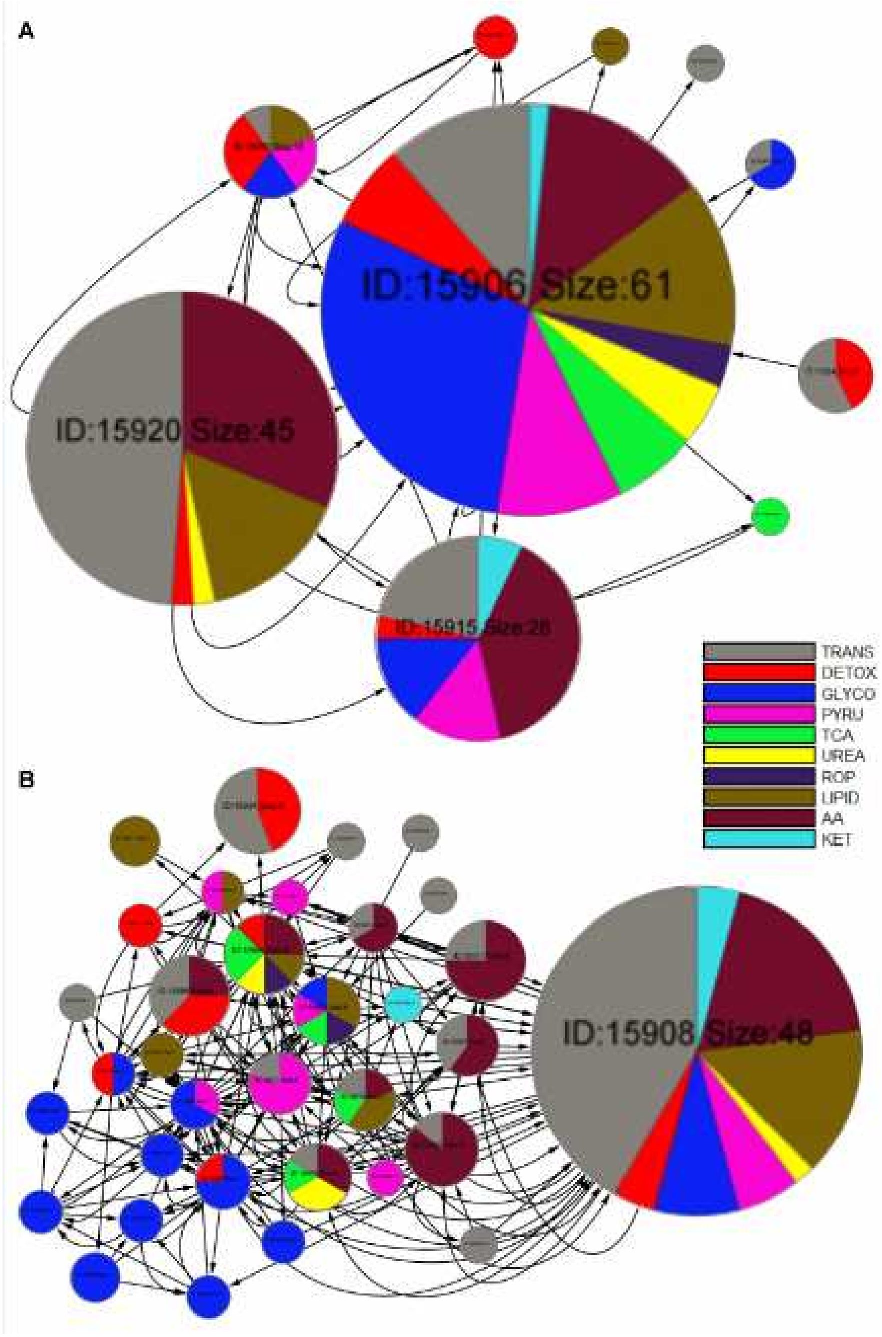


Figure 2-13:Network of Terminal Modules for Hepatocyte Model Partitions

(A) Newman's Partition. (B) ShReD-Based Partitioning. The interactions between modules represent interactions between reactions in the respective modules. The size of a module is proportional to the number of reactions in the module.

2.4.3 Impact of Allosteric Regulation

The impact of metabolic regulation on ShReD-based modularity was investigated by comparing the partitions for the metabolic network model with (Figure 2-11b) and without the allosteric interactions (Figure 2-12a). The two models yield qualitatively similar hierarchical partitions with subtle differences in the placement of reactions into modules (Appendix XX). These differences include the placement of reactions coupled to the pyruvate kinase reaction, which is subject to a high degree of allosteric regulation relative to other reactions in the network.

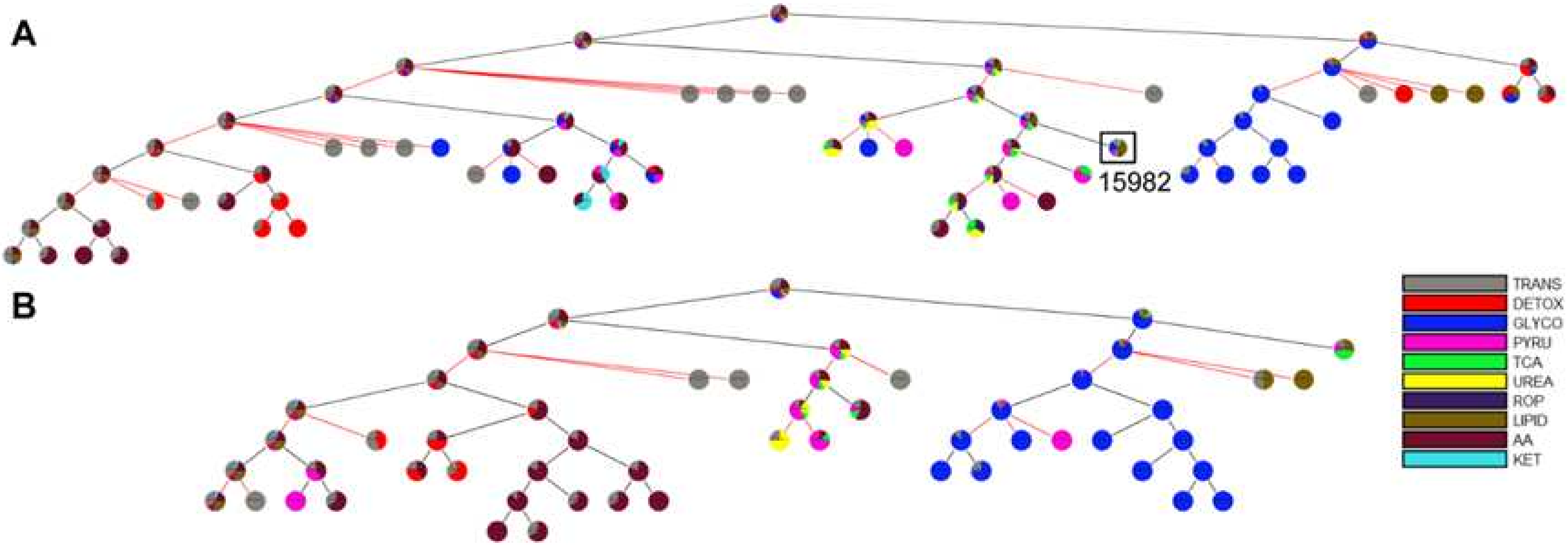


Figure 2-14: ShReD partitions of modified hepatocyte metabolic models

(A) Metabolic network with cofactors, but no regulatory edges. The boxed module (ID: 15982) contains pyruvate kinase. (B) Metabolic network with regulatory edges, but no cofactors. Note the absence of a redox module coupling detoxification reactions with lipid synthesis.

The quantitative impact of regulation is observed by comparing the number of ShReDs present in the network prior to the partition. At depth zero, there are approximately 250 additional ShReDs in the model with allosteric regulation compared to the model without regulation (Figure 2-15).

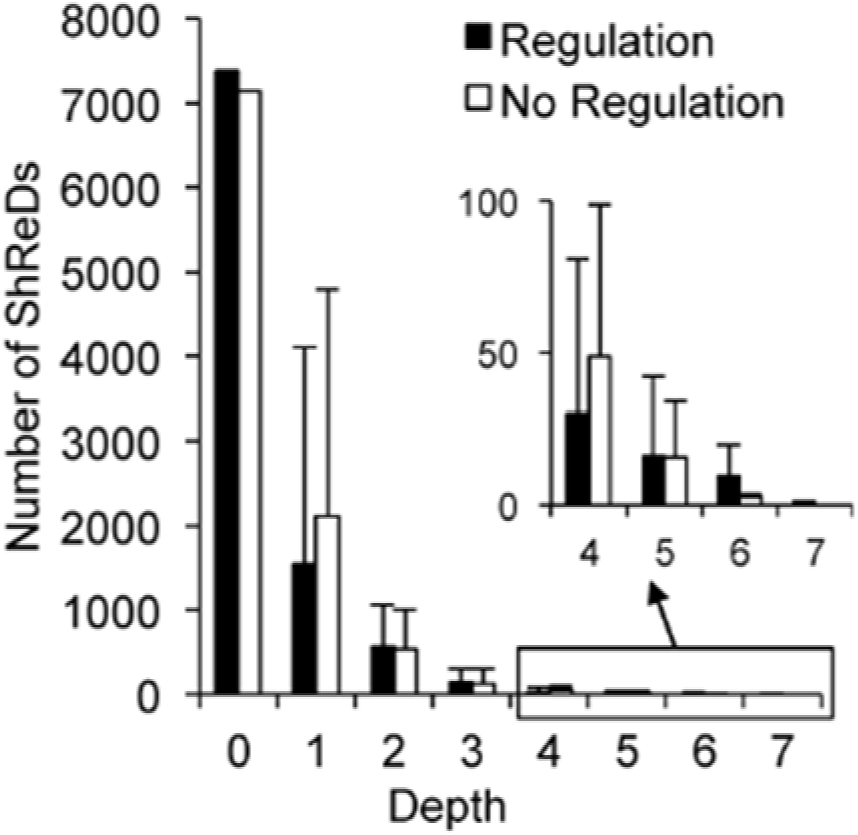


Figure 2-15: Effect of Regulation on Number of ShReDs with Respect to Depth

Average number of ShReDs in a module as a function of partition depth.

However, there is no obvious difference in the number of ShReDs between the two models at greater depths. There is also no obvious difference in the average ShReD at most depths, with the exception of depth zero, where the average ShReD is approximately 7 % shorter for the model with allosteric regulation compared to the model without regulation (Figure 2-16).

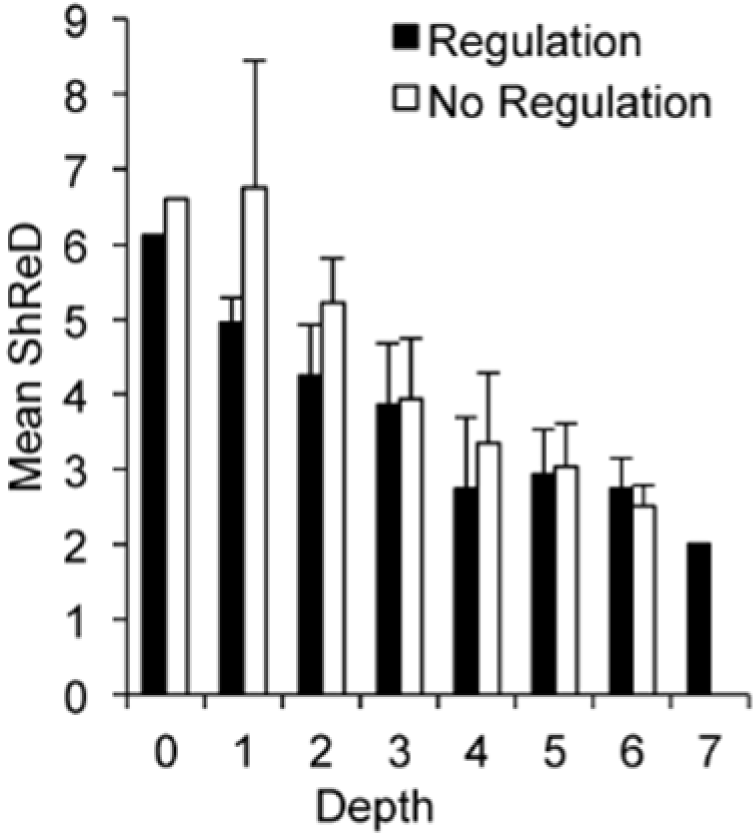
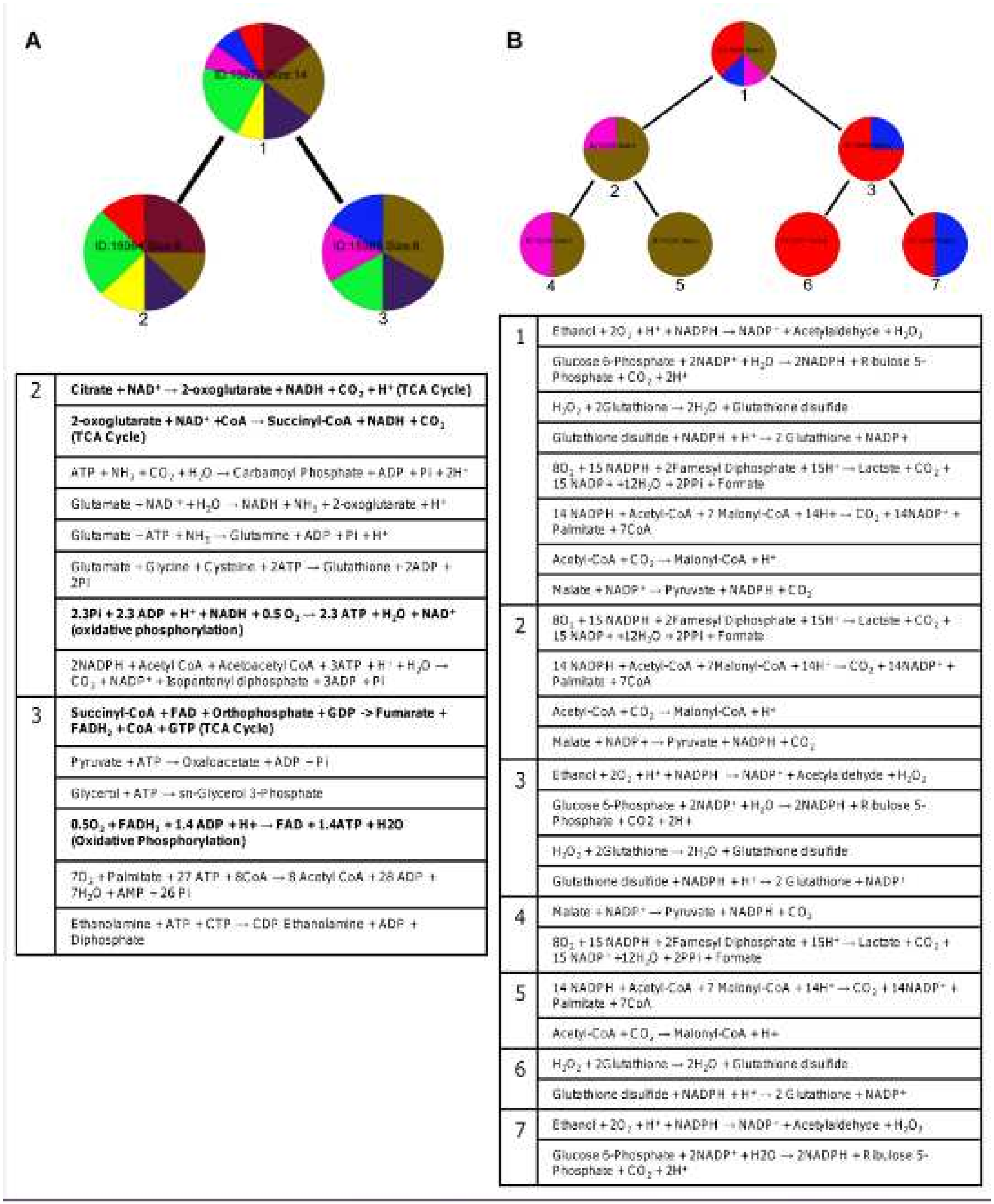


Figure 2-16: Effect of Regulation on the mean ShReD

Average ShReD of a module as a function of partition depth. Error bars represent one standard deviation.

2.4.4 Impact of Cofactors

We next assessed the impact of cofactors such as ATP, NADH, and NADPH on ShReD-based modularity by comparing the partition generated for the complete metabolic model (Figure 2-11b) to the partition for a partial model with regulatory edges, but lacking any interactions resulting from cofactors (Figure 2-14b). Qualitatively, the partitions reveal similar canonical groupings. Both partitions identify modules predominantly characterized by glucose metabolism (GLYCO) and modules predominantly characterized by amino acid metabolism (AA). Both partitions also group together reactions of the TCA cycle (TCA), urea cycle (UREA) and pyruvate metabolism (PYRU). A major difference between the two partitions involves the reactions of lipid metabolism (LIPID) and detoxification (DETOX). For the complete model, the ShReD partition identifies a module consisting of reactions from LIPID, DETOX, GLYCO, and PYRU (Figures 2-11b and 2-17b: ID: 15995), whereas no analogous module is identified for the model without cofactors. The reactions of module 15995 either produce or consume NADPH to support detoxification and lipid synthesis (Figure 2-17b).



2	**Citrate + NAD^+ → 2-oxoglutarate + NADH + CO_2 + H^+ (TCA Cycle)**
	2-oxoglutarate + NAD^+ +CoA → Succinyl-CoA + NADH + CO_2 (TCA Cycle)
	ATP + NH_3 + CO_2 + H_2O → Carbamoyl Phosphate + ADP + Pi + $2H^+$
	Glutamate + NAD^+ + H_2O → NADH + NH_3 + 2-oxoglutarate + H^+
	Glutamate + ATP + NH_3 → Glutamine + ADP + Pi + H^+
	Glutamate + Glycine + Cystine + 2ATP → Glutathione + 2ADP + 2Pi
	2.3Pi + 2.3 ADP + H^+ + NADH + 0.5 O_2 → 2.3 ATP + H_2O + NAD^+ (oxidative phosphorylation)
	2NADPH + Acetyl CoA + Acetoacetyl CoA + 3ATP + H^+ + H_2O → CO_2 + $NADP^+$ + Isopentenyl diphosphate + 3ADP + Pi
3	**Succinyl-CoA + FAD + Orthophosphate + GDP -> Fumarate + $FADH_2$ + CoA + GTP (TCA Cycle)**
	Pyruvate + ATP → Oxaloacetate + ADP + Pi
	Glycerol + ATP → sn-Glycerol 3-Phosphate
	$0.5O_2$ + $FADH_2$ + 1.4 ADP + H+ → FAD + 1.4ATP + H2O (Oxidative Phosphorylation)
	$7O_2$ + Palmitate + 27 ATP + 8CoA → 8 Acetyl CoA + 28 ADP + $7H_2O$ + AMP + 26 Pi
	Ethanolamine + ATP + CTP → CDP Ethanolamine + ADP + Diphosphate

1	Ethanol + $2O_2$ + H^+ + NADPH → $NADP^+$ + Acetylaldehyce + H_2O_2
	Glucose 6-Phosphate + $2NADP^+$ + H_2O → 2NADPH + Ribulose 5-Phosphate + CO_2 + $2H^+$
	H_2O_2 + 2Glutathione → $2H_2O$ + Glutathione disulfide
	Glutathione disulfide + NADPH + H^+ → 2 Glutathione + NADP+
	$8O_2$ + 15 NADPH + 2Farnesyl Diphosphate + $15H^+$ → Lactate + CO_2 + 15 NADP+ +$12H_2O$ + 2PPi + Formate
	14 NADPH + Acetyl-CoA + 7 Malonyl-CoA + 14H+ → CO_2 + $14NADP^+$ + Palmitate + 7CoA
	Acetyl-CoA + CO_2 → Malonyl-CoA + H^+
	Malate + $NADP^+$ → Pyruvate + NADPH + CO_2
2	$8O_2$ + 15 NADPH + 2Farnesyl Diphosphate + $15H^+$ → Lactate + CO_2 + 15 NADP+ +$12H_2O$ + 2PPi + Formate
	14 NADPH + Acetyl-CoA + 7Malonyl-CoA + $14H^+$ → CO_2 + $14NADP^+$ + Palmitate + 7CoA
	Acetyl-CoA + CO_2 → Malonyl-CoA + H^+
	Malate + NADP+ → Pyruvate + NADPH + CO_2
3	Ethanol + $2O_2$ + H^+ + NADPH → $NADP^+$ + Acetylaldehyde + H_2O_2
	Glucose 6-Phosphate + $2NADP^+$ + H_2O → 2NADPH + Ribulose 5-Phosphate + CO2 + 2H+
	H_2O_2 + 2Glutathione → $2H_2O$ + Glutathione disulfide
	Glutathione disulfide + NADPH + H^+ → 2 Glutathione + $NADP^+$
4	Malate + $NADP^+$ → Pyruvate + NADPH + CO_2
	$8O_2$ + 15 NADPH + 2Farnesyl Diphosphate + $15H^+$ → Lactate + CO_2 + 15 $NADP^+$ +$12H_2O$ + 2PPi + Formate
5	14 NADPH + Acetyl-CoA + 7 Malonyl-CoA + $14H^+$ → CO_2 + $14NADP^+$ + Palmitate + 7CoA
	Acetyl-CoA + CO_2 → Malonyl-CoA + H+
6	H_2O_2 + 2Glutathione → $2H_2O$ + Glutathione disulfide
	Glutathione disulfide + NADPH + H^+ → 2 Glutathione + $NADP^+$
7	Ethanol + $2O_2$ + H^+ + NADPH → $NADP^+$ + Acetylaldehyce + H_2O_2
	Glucose 6-Phosphate + $2NADP^+$ + H2O → 2NADPH + Ribulose 5-Phosphate + CO_2 + $2H^+$

Figure 2-17: Redox modules from ShReD-based partition of the hepatocyte model

Detailed composition of modules boxed in Figure 5b. (A) Coupled reactions of the TCA cycle and oxidative phosphorylation (highlighted in bold type) metabolizing NADH and $FADH_2$. (B) Coupled reactions metabolizing NADPH.

Quantitatively, the number of ShReDs trends lower when the cofactors are absent, with the largest difference observed at zero depth (Figure 2-18).

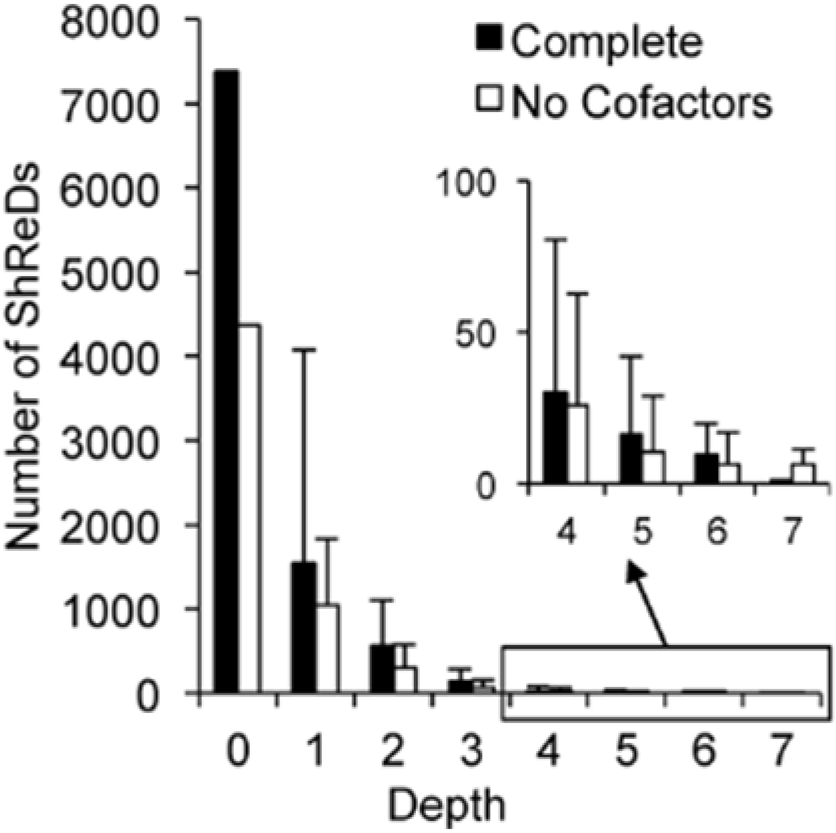


Figure 2-18: Partition Comparison between complete model and model with no cofactors: Number of ShReDs

Conversely, the average ShReD of a module is generally larger when the cofactors are absent, with the largest difference again observed at zero depth (Figure 2-19). At greater depths (> 3), the average ShReD plateaus to a value between 2 and 3 edges for both models.

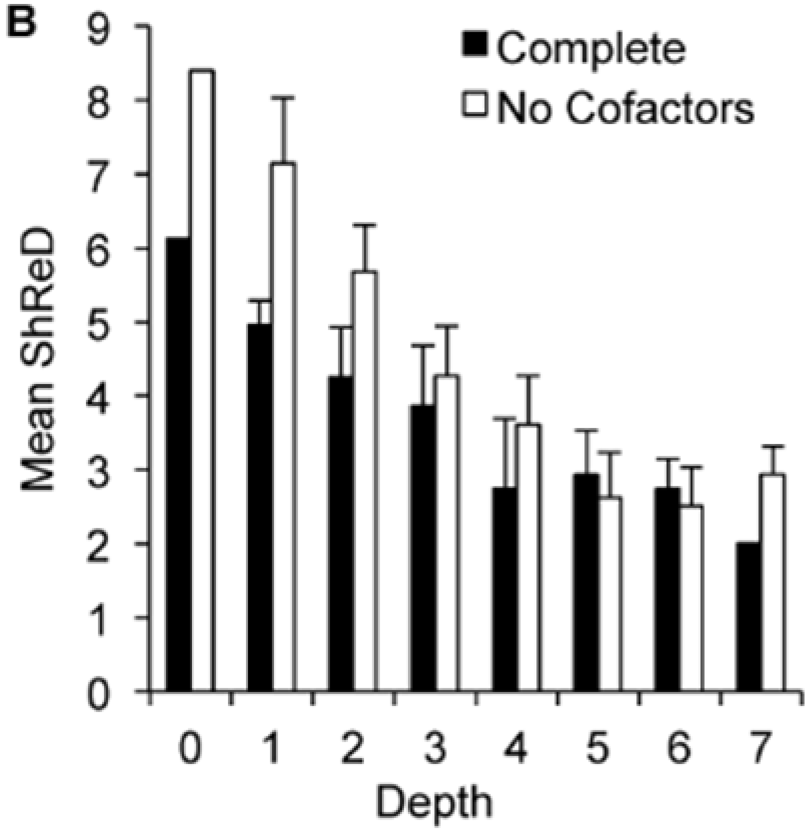


Figure 2-19: Partition Comparison between complete model and model with no cofactors: Mean ShReD

2.4.5 Comparison With Local Retroactivity

For completeness sake, we compared the partitions based on ShReD with partitions based on local, or nearest neighbor, retroactivity. To obtain local retroactivity partitions, the size of cycles was restricted to two edges, effectively eliminating all retroactive paths involving non-neighboring vertices. Algorithmically, $ShReD_{ij}$ was set to infinity, if $ShReD_{ij}$ was greater than 2. Biochemically, a locally retroactive interaction represented either a reversible reaction catalyzed by a single enzyme or two irreversible reactions with opposite stoichiometry. For all cases, including the EGFR signaling network as well as various versions of the hepatocyte metabolic network, partitions based on local retroactivity failed to generate any modules.

2.5 Discussion

In this work, we introduce the use of ShReD as a round trip distance metric, which can be combined with a partition algorithm (adapted from Newman's earlier work on community detection) to systematically identify biochemical reaction modules that feature cyclical interactions. The notion of grouping together network components based on "retroactivity" was first proposed by Saez-Rodriguez and coworkers, who hypothesized that a strictly downstream component should have little impact on the activity of an upstream component unless there is a feedback or retroactive relationship (Saez-Rodriguez *et al.*, 2008). It has been suggested that such feedback relationships contribute to robustness with respect to external perturbation, notably in signal transduction networks (Kitano, 2004). The ShReD metric accounts for cyclical interactions that span multiple reaction steps, and thus significantly extends on the prior work on retroactivity, which focused on local interactions between neighboring components. Previously, (shortest) path lengths between network components have been used to identify reaction modules by clustering, but without consideration of directionality and retroactivity (Ma *et al.*, 2004).

To evaluate the performance of ShReD as a module-detection metric, we performed two sets of comparisons. One set of comparisons involved the community detection algorithm presented by Newman, which also formed the basis for our partitioning algorithm. Newman's original algorithm partitioned based on connectivity, and favored the placement of a pair of network elements (vertices in the graph representation) into the same module if the number of connections between the two elements exceeded the expected (e.g. average) number of connections assuming an equivalent network with edges placed at random. The second set of comparisons involved the special case of local feedback loops or cycles arising from reversible reactions. The results of these comparisons were used to investigate how multi-step signaling loops or metabolic cycles, as opposed to conventional connectivity or reaction reversibility, contribute to the modular organization of biochemical networks.

Applied to a model network of EGFR signaling, the ShReD-based partitions generated modules with a greater number of cyclical interactions across all depths compared to Newman's connectivity-based partitions (Figure 2-10), consistent with the premise of the ShReD metric. Our results suggest that the total number of cyclical interactions in a network or module at least partially dictates the hierarchical depth of ShReD-based partitions. The ShReD-based partitions of the EGFR model generated one large terminal module with 99 reactions (Figure 2-6b, ID 22219), which could not be further modularized due to the relatively small number of cycles (a total of 10) in the module (~0.5 ShReDs per reaction). In contrast, the GPCR dominated module (ID 22203) has 167 cycles and 774 ShReDs connecting just 39 reactions (~20 ShReDs per reaction), and can be further partitioned to generate 4 additional levels of hierarchy.

In the case of the liver metabolic network, which has a substantially greater number of cycles (arising from allosteric feedback loops) compared to the EGFR signaling network, the difference between ShReD and Newman partitions is more dramatic. ShRed partitions again lead to greater hierarchy, reaching a depth of 7, whereas Newman's partition only reaches a depth of 3. The greater hierarchy achieved using the ShReD metric is significant, because the partition algorithm is essentially identical to Newman's algorithm, i.e. the only difference is the metric used to calculate the modularity score *Q*. For both metrics, a module is further partitioned only if the *Q* score is positive after the partition. Indeed, the scoring criterion based on the ShReD metric is actually more stringent, because the algorithm performs an additional test to ensure that the modules resulting from a partition each has at least one cycle. In this regard, the greater hierarchy generated by the ShReD (as opposed to the partition algorithm) gives credence to the metric for being able to identify hierarchical modules based on the preservation of cycles.

The retroactive interactions captured by ShReD include not only reaction reversibility (as in previous work (Saez-Rodriguez *et al.*, 2008)), but also cycles and feedback loops involving multiple reactions and allosteric effectors. Feedback loops resulting from allosteric regulation of an upstream enzyme by a downstream product represent an important regulatory motif that is common to biochemical networks. To examine the impact of feedback loops on modularity, ShReD partitions were obtained for the metabolic network with and without allosteric regulation. While qualitatively similar, the partitions differed in the placement of highly regulated reactions. For example, biochemistry textbooks generally associate pyruvate kinase (PK) with glycolysis, where the enzyme catalyzes the terminal step. The enzyme's activity is subject to allosteric regulation by several sugar phosphates produced upstream in glycolysis. On the other hand, the enzyme's product, pyruvate, is highly connected to the TCA cycle and amino acid pathways through anaplerosis and transamination reactions. When regulatory edges are absent, ShReD partitions place PK in one of the terminal leaf nodes along with reactions of lipid metabolism, pyruvate metabolism, the TCA cycle, oxidative phosphorylation and ketone body synthesis (Figure 2-14a, ID: 15982). When regulatory edges are present in the model, however, the partitions place PK in a module dominated by reactions of sugar metabolism (Figure 2-11b, ID 15939), consistent with textbook biochemistry. In this regard, the ShReD metric captures the impact of both stoichiometric connectivity and feedback regulation in determining modularity.

As many of the allosteric regulators were energy currency metabolites, we also examined the partitions for a partial metabolic model that lacks these cofactors. The resulting network contains fewer ShReDs, presumably reflecting an overall decrease in the total number of paths.

Compared to the complete model, the corresponding ShReDs (connecting the same reaction vertices) of the partial model are ~30% *longer*, indicating that allosteric feedback and other cofactor-dependent interactions more tightly couple the reactions in the network. In the present study, abstracting the metabolic network as a reaction-centric graph greatly facilitated the inclusion of cofactors in the modularity analysis, identifying both intuitive and non-canonical groupings that could not be identified by removing interactions effected by cofactors. For example, not including the cofactors in the model would completely isolate the oxidative phosphorylation reactions and carbamoyl phosphate production reaction from the rest of the metabolic network as disconnected components. Including the cofactors allows these reactions to be placed into modules; for the complete metabolic model, these reactions are kept together at a height of 2. Another example of cofactor-dependent modularity involves the association of NADH and $FADH_2$ oxidation with different reactions in and around the TCA cycle (Figure 2-17a**)**. The partitions place NADH oxidation into a module (ID: 15984) that also contains isocitrate and alpha-keto glutarate dehydrogenases, which are NADH producing reactions in the TCA cycle. Similarly, $FADH_2$ oxidation is placed in a module (ID: 15985) containing succinate dehydrogenase, which reduces FAD^+ to $FADH_2$. The coupling between TCA cycle reactions and oxidative phosphorylation is intuitive. However, the TCA cycle reactions are also highly connected to reactions in glutamate metabolism and β-oxidation, associations that may be subjectively less intuitive. In this light, ShReD partitions reflect an emphasis on cyclical interactions mediated by the cofactors. A third example of an intuitive, yet non-canonical grouping involves the drug transformation reactions. In the present study, the metabolic model included reactions that are induced by troglitazone, a hydrophobic anti-diabetic compound withdrawn from the market due to severe hepaotoxicity. Module 15995 illustrates the cyclical interactions coordinating reactions of several different canonical pathways, including glutathione, lipid, glucose, and pyruvate metabolism (Figure 2-17b). A dominant characteristic (exhibited by seven of the nine reactions) of this module is the production and consumption of NADPH, again underscoring the significance of the cofactors in determining the modularity.

To examine whether the influence of the cofactors reflected the relatively small size of the model network (comprising ca. 150 reactions), we also applied the ShReD-based modularity analysis to a larger model of the human liver (comprising ca. 2500 reactions) [21]. The analysis again identified cofactor modules centered on NADH and NADPH consumption and production, similar to the smaller liver model. Many of the terminal modules for the larger model comprised reactions that were grouped into analogous modules for the smaller model, suggesting that the size of the model did not qualitatively alter the structural organization of the metabolic network. Quantitatively, the maximum hierarchical depth was greater for the larger network, increasing from 7 to 16. The increased depth was presumably due to the greater detail of the HepatoNet1 model, which includes many additional pathways of amino acid, lipid and nucleotide metabolism.

2.6 Conclusion

In this chapter, we present a novel methodology for modularity analysis that enables hierarchical partitions of biochemical networks by preserving feedback loops and other cyclical interactions. To the best of our knowledge, the present study is the first to build a module detection method that focuses on cycles or feedback loops as the key structural feature. The present study is also the first to account for cofactors in modularity analysis, further emphasizing the role of pathway regulation in network modularity. Previously, studies on modularity have generally ignored cofactors, citing methodological challenges arising from having to place these highly connected hub metabolites into particular modules [20-22]. It should be noted that the current analysis, which does not weight the edges in calculating the ShReDs, implicitly assumes that all reactions in the network are equally engaged. Clearly, the levels of engagement can be expected to vary across different reactions, and should ideally be weighted appropriately, by using quantitative activity data such as metabolic flux. For example, a high glycolytic flux may confer a larger weight to edges representing PK regulation, which in turn may impact the overall modularity of the network. Moreover, cells subjected to different chemical or genetic perturbations will likely exhibit different flux dynamics, which would need to be reflected in the metric to obtain partitions that meaningfully analyze the modularity of a dynamic system such as the biological cell. A thorough examination of the role of reaction engagements in modularity analysis is discussed in Chapter 4.

3 Using ShReD-Based Hierarchical Modularity to Identify Cyclical Elementary Flux Modes

3.1 Abstract

The hierarchical modularity of metabolic networks provides a framework for understanding the functional organization of cellular metabolism. Systematic modularity analysis of networks systematically uncovers reaction groupings that may not be intuitive based on a two-dimensional cartography of metabolism. In the previous chapter, the ShReD-based partition algorithm was proposed to discover modules that are enriched in cyclical interactions that may offer local robustness to perturbations. In this chapter, we seek to identify substrate (futile) cycles within ShReD-based modules by searching for cyclical elementary flux modes (EFM). Substrate cycles have been discussed in literature as promising bioenergetics targets for disorders such as obesity, and were shown in this introduction to confer local robustness. Identification of novel substrate cycles in large scale networks (>1000 reactions) is challenging, so applying quantitative tools to discover novel substrate cycles would be invaluable. Since an exhaustive set of EFMs for a large scale network is computationally limiting, we focus on a targeted subset of reactions for the motif search. Here, we report that our methodology is able to identify substrate cycles in ShReD-based modules for the hepatonet1 model, a large scale liver model. While prior work only focused on identifying substrate cycles that consume ATP, we report potential substrate cycles that have a net production or consumption of a wide range of cofactors and hub metabolites.

3.2 Introduction

Graph-based modeling of metabolism has shown that whole-cell or tissue-specific metabolic networks may be functionally organized into hierarchical modules (M. E. J. Newman, 2006; Ravasz *et al.*, 2002). Modularity analysis allows for the identification of interactions between reactions that may not necessarily be intuitive from a two-dimensional cartography of metabolism, such as the one presented in the KEGG atlas (Minoru Kanehisa *et al.*, 2010). For example, we recently introduced the Shortest Retroactive Distance (ShReD) as a metric to capture distant cyclical interactions among reactions in a network to uncover modules enriched in feedback loops and metabolic cycles that spanned several canonical textbook pathways, thus revealing unexpected reaction groupings (Sridharan *et al.*, 2011). Such a systematic hierarchical partitioning of the network allows an investigator to then survey the local topology of the network at a desired modular resolution to identify important motifs in the context of the module's biochemical function. Since motif searches on large scale networks are computationally expensive (Grochow and Kellis, 2007), systematic approaches to focus the search towards a local neighborhood would be very valuable.

One motif of particular importance is the metabolic substrate cycle, which can be described as a set of reactions that serve to replenish any metabolite participating in the cycle without a net catabolic or anabolic function. These cycles would be thermodynamically infeasible without the involvement of a metabolic cofactor, or ion exchange across membranes. The most commonly known substrate cycles involve reactions from glycolysis and gluconeogenesis that are simultaneously active: inter-conversion of glucose and glucose 6-phosphate; fructose 6-phosphate and fructose 1,6-bisphosphate; and phosphoenolpyruvate and pyruvate. Once thought to be futile, and hence the synonymous terminology in literature, these cycles have more recently been ascribed roles in cellular functions such as thermogenesis (Reidy and Weber, 2002) and metabolic control (Sazanov and Jackson, 1994). Substrate cycles can maintain an independent steady-state cycle flux, which can in theory fluctuate without altering other fluxes in the metabolic network provided the cycle has minimal effect on cofactor pools. This feature promotes local robustness to external perturbations, which is an important property of modularity (Stelling *et al.*, 2004). Conversely, if a substrate cycle quantitatively impacts the consumption or generation of a cofactor, then the enzymes of the substrate cycle could be up-regulated or down-regulated to selectively adjust the cofactor level. In fact, a substrate cycle in adipose tissue, namely the esterification and hydrolysis of triglycerides, have been investigated as bioenergetics targets for the treatment of obesity (Tseng *et al.*, 2010). Differential expression of substrate cycle enzymes has also been explored as a possible approach to alter cancer cell metabolism (Locasale and Cantley, 2010).

Applied to large-scale model reconstructions of metabolic networks, systematic identification of the cycle motif using computational tools can lead to the discovery of novel substrate cycles. Gebauer and coworkers recently pointed out that substrate cycles and cyclical elementary flux modes (EFMs) are equivalent, and showed that the latter can be identified by applying existing EFM enumeration algorithms, provided that the model network has been appropriately reduced to remove exchange reactions and associated metabolites (Gebauer *et al.*, 2012). Using the EFMEvolver algorithm (Grosse *et al.*, 2009), which is based on a genetic algorithm, they enumerated over 200,000 EFMs in the cytosol with a medium length (defined as the number of reaction steps in the cyclical EFM) of 35. However, the cell's physiology may or may not be able to coordinate the operation and regulation of substrate cycles spanning a large number of reaction steps. Moreover, substrate cycles that span a large number of reaction steps across several metabolic modules may not be as amenable for experimental intervention, since alteration of enzyme activities would be required for multiple metabolic pathways. Since a complete and exhaustive EFM enumeration on large scale networks is not yet computationally feasible, this approach will also inevitably preclude the identification of some substrate cycles that may be more relevant in the context of modularity and metabolic function. For example, the prior work limited the EFM search to only those substrate cycles involving ATP consumption. However, many previously identified substrate cycles involve NADH, NADPH or ion transfer across membranes.

Therefore, we propose an alternative approach for the computational identification of potential substrate cycles in the context of the hierarchical modularity of the metabolic network. In this study, we apply ShReD-based partitioning on a large scale liver network (hepatonet1) (Gille *et al.*, 2010), and conduct an *exhaustive* EFM analysis on individual modules rather than the whole network using EFMtool (Terzer and Stelling, 2008). Since ShReD-based modules are identified based on cyclical interactions, they should in principle be conducive to identifying substrate cycles. The EFM search is exhaustive for all but a small number of modules at the top of the hierarchy, which contained too many reactions for EFMtool to complete the enumeration within a reasonable time. In this manner, the discovered cyclical EFMs are placed in the context of the overall functional organization of the network. We identify cyclical EFMs involved in transport, lipid synthesis, folate metabolism, sugar metabolism, and amino acid metabolism, with the net transformation of several different cofactors including NADH, NADPH, as well as other hub metabolites such as sulfate, phosphate, and hydronium ions.

3.3 Methods

3.3.1 Substrate Cycles and Model Representation

The model system in Figure 3-1A shows a substrate cycle between two reactions [R_2,R_4] operating at the expense of a reduced cofactor ER. Here, the set [R_2,R_4] is also a cyclical EFM, if the cofactor is not included in the balance. However, for the system described in Figure 3-1B, the same set of reactions do not constitute a substrate cycle because R_2 also produces M_5, and the cycle [R_2,R_4] thus has a net catabolic function on substrate M1. In this case, [R_2,R_4] is not a cyclical EFM. These model systems can be abstracted as reaction-centric graphs, where an edge is drawn from a reaction R_i to another reaction R_j if a metabolite produced by the first reaction R_i is consumed by the second reaction R_j. Based on this scheme, the model systems shown in Figure 3-1A and Figure 3-1B can be represented by identical reaction-centric graphs (Figure 3-1C). This example illustrates that while all substrate cycles can be represented as a directed cycle in a reaction centric graph, not all directed cycles necessarily represent substrate cycles. Consequently, it is necessary to perform an additional check to determine whether a directed cycle in a reaction graph also constitutes a cyclical EFM.

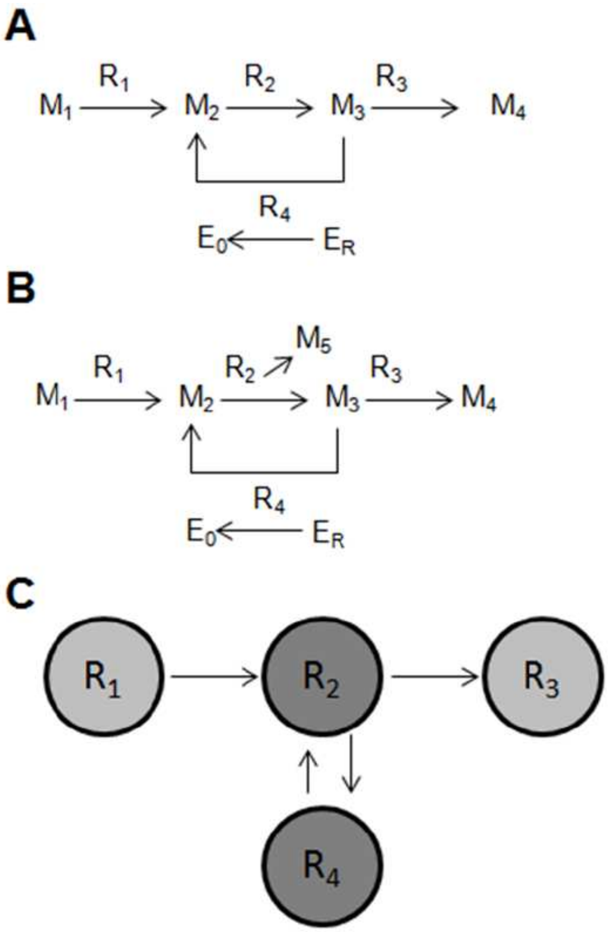


Figure 3-1: Substrate cycles and model representation

(A) Substrate cycle involving R2 and R4 at the expense of E_R. (B) This system does not possess a substrate cycle because of the net production of M_5. (C) Both systems (A) and (B) can be represented by the same reaction-centric graph.

However, the same set of reactions for the system described in Figure 3-1B does not constitute a substrate cycle because R_2 also produces M_5, and the cycle $[R_2,R_4]$ has a net catabolic function on substrate M_1. In this case, $[R_2,R_4]$ is not a cyclical EFM. Both of these systems can be abstracted as the same reaction-centric graph where if a metabolite produced by R_i is consumed by reaction Rj, then an edge is drawn from Ri and Rj (Figure 3-1C). It follows that all substrate cycles will be represented as a directed cycle in a reaction centric graph. However, not all directed cycles denote substrate cycles, and one has to check if the reactions constitute a cyclical EFM first.

3.3.1.1 Hepatonet1 Model

The hepatonet1 model (Gille *et al.*, 2010), a large scale model of liver metabolism, was downloaded in SBML format and converted to a stoichiometric matrix (S-matrix). From this model, two sub-models were constructed, one for ShReD-based partitioning (hepShReD), and one for EFM analysis (hepEFM). For the hepShReD model, inorganic hub compounds such as H2O and H^+ were removed, as they are involved in a majority of metabolic reactions and thus define reaction interactions only in a generic manner. However, energy cofactors such as ATP, NADH, and NADPH were kept, as they mediate important regulatory interactions. The hepShReD model was then abstracted as a reaction-centric graph. Reaction nodes that either produce or consume an extracellular metabolite were removed from the network. The resulting network comprised of 1418 reactions. The hepEFM model was constructed by removing all hub and cofactor metabolites from the original hepatonet1 model. Removing these metabolites was necessary to compute substrate cycles using EFM analysis, as cofactors should not be balanced in a substrate cycle. Similar to the hepShReD model, reactions producing or consuming extracellular metabolites were also removed.

3.3.2 ShReD-based Partitioning of Network

The hepShReD graph was iteratively partitioned based on the ShReD-metric as described in our previous work (Sridharan *et al.*, 2011). Briefly, a ShReD value (length of the shortest directed cycle spanning two reaction nodes) was computed for every reaction pair in a given sub-network, starting from the parent network comprising of all 1418 reactions. A binary partition was made for each sub-network to maximize a modularity score such that reactions that have a strong cyclical interaction, as determined by short ShReD values, are placed together into the same module. If a partitioned sub-network was not completely connected, it was broken down into its connected components. The result was a hierarchical tree of modules, each comprising a subset of reactions from the original 1418 reactions.

3.3.3 Identification of Cyclical EFMs within Modules

To identify and enumerate all the EFMs within each module, the S-matrix corresponding only to those reactions within the module was extracted from the hepEFM S-matrix, to which EFMTool (Terzer and Stelling, 2008) was applied. To check whether a modular EFM was also a cycle, the reactions involved in a particular EFM were abstracted as a reaction-centric sub-graph. We then determined, using depth first search (DFS), whether there existed a set of strongly connected components (SCCs) involving all the vertices. SCCs describe a group of nodes where each node can reach every other node via a directed path. In this case, the existence of a set of SCCs equates to the presence a substrate cycle, where each node is only traversed once in the cyclic path. There are no cases in which set of SCCs would be identified where a reaction node traversed more than once as in Figure 3-2, because such a set would not have been identified as an EFM. In the following example, subsets [R_i,R_j] and [R_j,R_k] each constitute a cyclical EFM on their own.

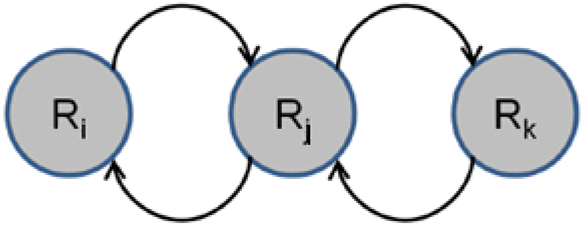


Figure 3-2: Example of directed cycle that is not a cyclical EFM

All modules in the hierarchical tree of partitions for hepShReD were sorted by height (defined as the maximum path length from the specified module to a terminal node). Modular EFM analysis, as described above, was performed on each module in the hierarchical partition tree in order of increasing height. Modules at lower height contain fewer reactions, and EFMTool is more likely to complete the exhaustive search. For modules higher up in the hierarchy, if EFMTool was unable to complete the search within one hour, the program moved on to the next module. Searching for modular EFMs using this ‘bottom up’ approach starting from the terminal modules is preferable to starting from the root parent node coming ‘top down’, which could take longer to complete by spending an hour on each module towards the top.

3.3.4 Net Consumption of Cofactors and Hub Metabolites

Once a cyclical EFM was identified as a potential substrate cycle, the stoichiometric matrix for the original network model was used to determine which cofactors and hub metabolites were net consumed or produced for the cycle. This was done for all unique substrate cycles, and a tally was kept for how many cyclical EFMs each cofactor was either net consumed or produced.

3.4 Results

3.4.1 ShReD-based Partition of Hepatonet1 Model

The ShReD-based hierarchical partitioning of the hepShReD sub-model yields 2098 modules, each of which comprises a subset of the original 1418 intracellular reactions (Figure 3-3). The parent module contains all 1418 reactions, but it is not completely connected and is broken into its connected components. A ShReD-based binary partition is then performed on the largest connected component, which divides reactions between those contained in the modules denoted by Figure 3-3A and the rest of the network. After each binary partition, reaction pairs within each sub-network have shorter ShReD values on average than do reaction pairs across the two partitioned sub-networks. In this manner, reactions that are tightly coupled through cyclical interactions are identified and reactions that remain together at further depths in the hierarchy are in theory most functionally related. Only nine modules close to the parent node in the hierarchy contained too many reactions for EFMtool to complete. Interestingly, the number of cyclical EFMs within modules did not correlate with the size of the module, in that the density of cyclical EFMs is far greater in some parts of the hierarchy. For example, module 143829 in Figure 3-3B contains 226,014 cyclical EFMs for 121 reactions (Table1). However, module 144976 in Figure 3-3A contains more reactions (139), but only 8 cyclical EFMs (Table1). Module 143829 also has the most number of cyclical EFMs of all modules for which EFMtool completed.

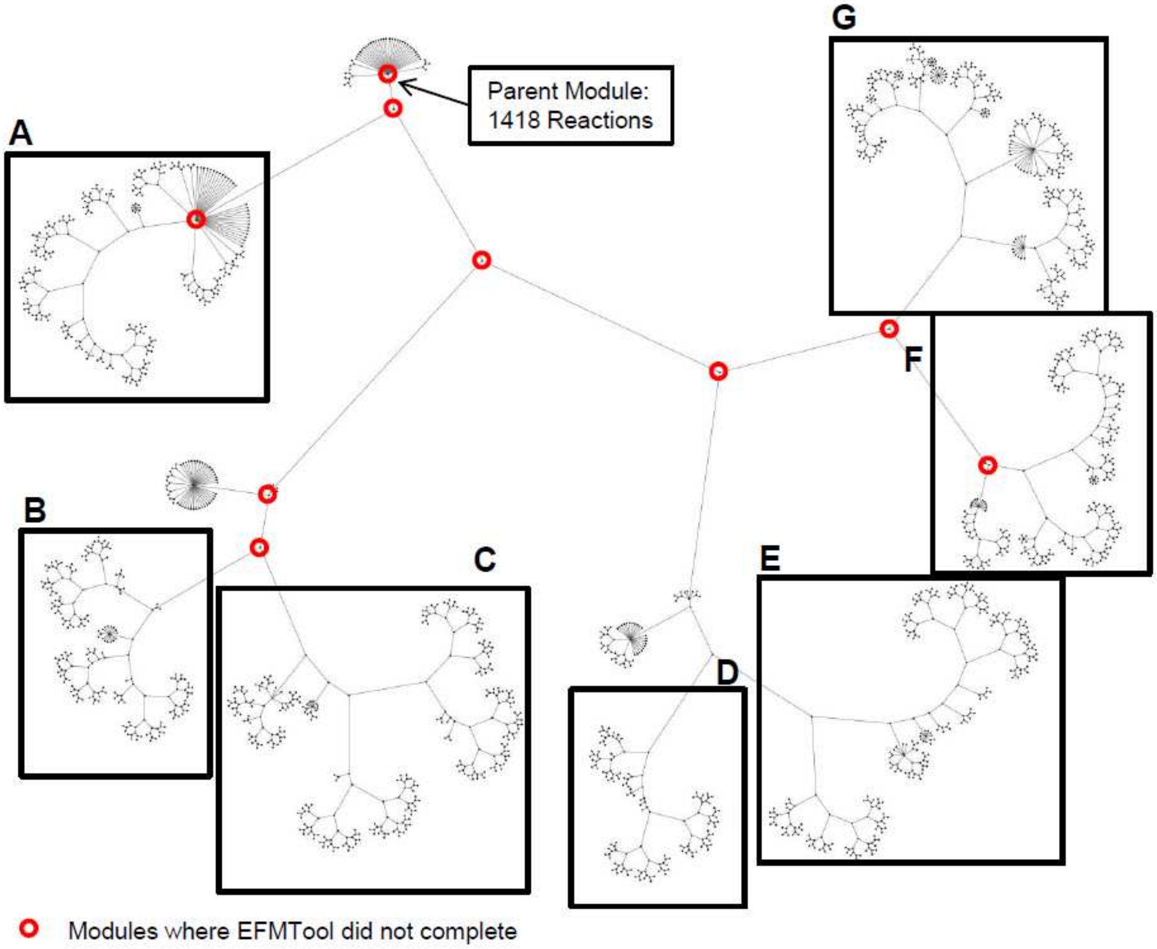


Figure 3-3: ShReD-based partition of hepatonet1 model

Table 1: Sample cyclical EFMs in modules

Module ID	Canonical Metabolic Functions Associated with Module	Sample Cyclical EFM in Module

144976: Fig 3A
N = 139
CE = 8
	- Cholesterol synthesis
	-VLDL synthesis
		R1136:4alpha-Methylzymosterol-4-carboxylate (r) + NADP+(r) ⟷ 3-Keto-4-methylzymosterol(r) + NADPH(r)
		R1137: NAD+(r) + 4alpha-Methylzymosterol-4-carboxylate (r) ⟷ NADH(r)+CO2(r)+ 3-Keto-4-methylzymosterol(r)

143830: Fig 3B
N = 121
CE = 226,014
	-TCA cycle
	- Mitochondria/Cytosol Transport
		R0829: Succinate(c) + Sulfate(m) ⟷ Succinate(m) + Sulfate(c)
		R0831: Malate(c) + Pi(m) ⟷ Malate(m) + Pi(c)
		R0931: Isocitrate(m) + Malate(c) ⟷ Isocitrate(c) + Malate(m)
		R0915: Citrate(c) + Succinate(m) ⟷Citrate(m) + Succinate(c)
		R0917: Citrate(c) + Isocitrate(m) ⟷ Citrate(m) + Isocitrate(c)

143829: Fig 3C
N = 182
CE = 20
	-Beta Oxidation
	-Glutamate/Proline Metabolism
	-Ketone Body Synthesis
	-TCA cycle
		R0223: 2-Methyl-3-oxopropanoate(m) + CoA(m) + NAD+(m) ⟶ Propanoyl-CoA(m) + CO2(m) + NADH(m)
		R0414: ATP(m) + Propanoyl-CoA(m) + HCO3-(m) ⟶ ADP(m) + Pi(m) + Methylmalonyl-CoA(m)
		R0571: Methylmalonyl-CoA(m) + H2O(m) ⟷ Methylmalonate(m) + CoA(m)
		R0643:2-Methyl-3-oxopropanoate(m) + NAD+(m) + H2O(m) ⟷ Methylmalonate(m) + NADH(m)

142886: Fig 3D
N = 78
CE = 2
	-NADH(c) production/consumption
	-Lactate Dehydrogenase
		R0267: CMP-N-acetylneuraminate(c) + O2(c) + NADH(c) ⟷ CMP-NeuNGc(c) + NAD+(c) + H2O(c)
		R0269: CTP(n) + N-Acetylneuraminate(n) ⟷ PPi(n) + CMP-N-acetylneuraminate(n)
		R0400: NAD+(c) + O2(c) + N-Acetylneuraminate(c) ⟷ NeuNGc(c) + NADH(c) + H2O(c)
		R0668: NeuNGc(c) + CTP(c) ⟷ PPi(c) + CMP-NeuNGc(c)
		R1461: CMP-N-acetylneuraminate(c) ⟷ CMP-N-acetylneuraminate(n)
		R1462: N-Acetylneuraminate(c) ⟷ N-Acetylneuraminate(n)

142887: Fig 3E
N = 151
CE = 307
	-Lipid Biosynthesis
	-Folate Metabolism
	-NADPH(c) production/consumption
		R0225: THF(c) + NADP+(c) ⟷ Dihydrofolate(c) + NADPH(c)
		R0227: 10-Formyl-THF(c) + H2O(c) + NADP+(c) ⟶ THF(c) + CO2(c) + NADPH(c)
		R0293: 5,10-Methylene-THF(c) + NADP+(c) ⟷ 5,10-Methenyl-THF(c) + NADPH(c)
		R0371: 5,10-Methenyl-THF(c) + H2O(c) ⟷ 10-Formyl-THF(c)
		R0501: dUMP(c) + 5,10-Methylene-THF(c) ⟷ Dihydrofolate(c) + dTMP(c)

142411: Fig 3F
N = 152
CE = 4
	-Acyl CoA activation in cytosol via carnitine
	-VLDL synthesis
		R0066: ATP(c) + Acetate(c) + CoA(c) ⟶ AMP(c) + PPi(c) + Acetyl-CoA(c)
		R0485: Glucosamine-6P(c) + Acetyl-CoA(c) ⟶ CoA(c) + N-Acetylglucosamine-6P(c)
		R0486: N-Acetylglucosamine-6P(c) + H2O(c) ⟷ Glucosamine-6P(c) + Acetate(c)

141811: Fig 3G
N = 359
CE = 575
	-Sugar Metabolism
	-Amino Acid Metabolism
	-Protein Synthesis
		R0487: Fructose-1,6PP(c) + H2O(c) ⟶ Fructose-6P(c) + Pi(c)
		R0736: R0736: ATP(c) + Fructose-6P(c) ⟶ ADP(c) + Fructose-1,6PP(c)

		R0129: GSH(c) + H2O(c) ⟷ Glutamate(c) + Cys-Gly(c) **R0131**: ATP(c) + gamma-Glutamyl-cysteine(c) + Glycine(c) ⟶ ADP(c) + Pi(c) + GSH(c) **R0212**: ATP(c) + Glutamate(c) + Cysteine(c) ⟶ ADP(c) + Pi(c) + gamma-Glutamyl-cysteine(c) **R0214**: H2O(c) + Cys-Gly(c) ⟷ Cysteine(c) + Glycine(c)

N = Number of reactions in module, CE = Number of Cyclical EFMs in module

3.4.2 Modular Functions and Representative Substrate Cycles

For each boxed group in Figure 3, the recognizable metabolic functions of the module with the most number of reactions in that group are listed along with representative cyclical EFMs in Table 1. We report the reactions as cyclical EFMs and not substrate cycles. While any substrate cycles can be represented as a cyclical EFM, not every cyclical EFM is necessarily an active substrate cycle, For module 144976 in Figure 3-3A, which comprises of reactions involved in cholesterol synthesis and very low density lipoprotein (VLDL) metabolism, we list a cyclical EFM that represents a common motif where two reversible reactions carry out the same biochemical transformation but are catalyzed by different cofactors. In this case, the conversion of 3-Keto-4-methylzymosterol(r) to 4alpha-Methylzymosterol-4-carboxylate(r) is mediated by NADPH/NADP in one reaction and NADH/NAD in the other. Substrate cycling in this case could result in a net transformation of NADPH to NADH or NADH to NADPH depending on the direction of the cycling.

Many cyclical EFMs involve transport reactions across membranes. In module 143829 (Figure 3-3C), many reactions involve the transport of TCA cycle intermediates such as citrate, malate, and succinate between the cytosol and mitochondria. Depending on the cycling direction, the function of such substrate cycles, if active, could be the net transfer of protons or phosphate ions across the mitochondrial membrane. These transport-based cyclical EFMs are contained within a module that also contains TCA cycle reactions, suggesting a relationship between energy production reactions and membrane charge transfer in order to mediate the proton gradient required for oxidative phosphorylation.

In some cases, the relationship between the function of certain cyclical EFMs and their modular placement is intuitive. For example, module 141811 (Figure 3-3G) comprises reactions in sugar metabolism, amino acid metabolism, and protein synthesis. Within this module, the cyclical EFM involving the inter-conversion of fructose-6-phosphate and fructose 1,6 bisphosphate is a step in both glycolysis and gluconeogenesis. Similarly, the cyclical EFM involving the production and degradation of glutathione at the expense of ATP directly influences the metabolism of glutamate, cysteine, and glycine, the three amino acids required for the synthesis of glutathione. However, we also found less intuitive associations between cyclical EFMs and biochemical modules. Many ShReD modules group together reactions that span several distinct textbook pathways based on the shared production and consumption of metabolic cofactors. For example, a cyclical EFM in module 142886 (Figure 3-3D) that involves N-Glycolyneuraminate (NeuNGc) metabolism belongs to the same module as lactate dehydrogenase. The close interaction between these reactions with ostensibly unrelated functions exists because of the shared production and consumption of cytosolic NADH. Similarly, a cyclical EFM in module 142887 (Figure 3-3E) captures the one carbon pool cycle in folate metabolism with a net production of NADPH, which directly connects to lipid biosynthesis and other reactions that require that cofactor.

3.4.3 Modular Cyclical EFM Length

For each module that contained at least one cyclical EFM, the mean cyclical EFM length (length being defined as the number of reactions that span the cycle) was computed and plotted against the module size (Figure 3-4). The mean cyclical EFM length for modules ranges from two (2) to nine (9). While there is a general trend that modules with more reactions have a greater mean cyclical EFM length, the correlation is not strong. For module 141811 (Figure 3-5), which contains the most number of reactions for which EFMtool was able to complete, the distribution of cyclical EFM length is shown (Figure 3-5). For this particular module, the distribution of lengths is bimodal with a large cluster of cyclical EFMs between lengths two (2) and five (5) and another cluster between length nine (9) and eleven (11).

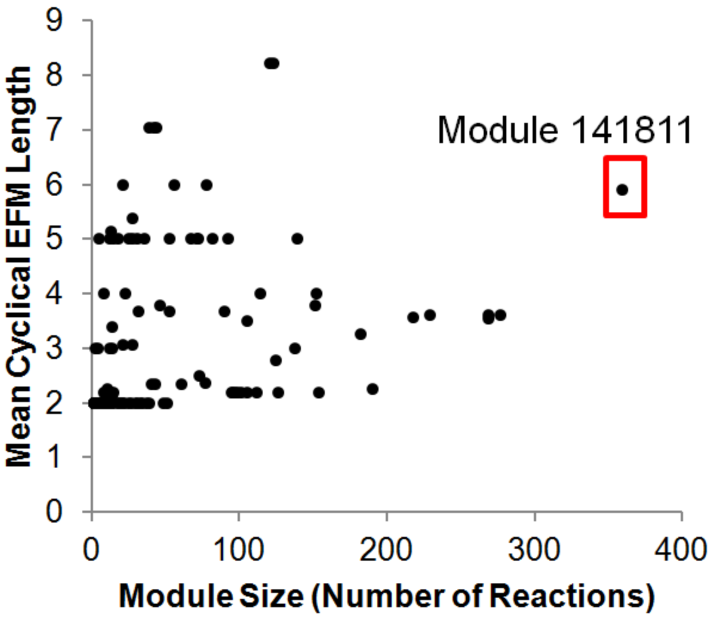


Figure 3-4: Mean cyclical EFM length vs. Module size

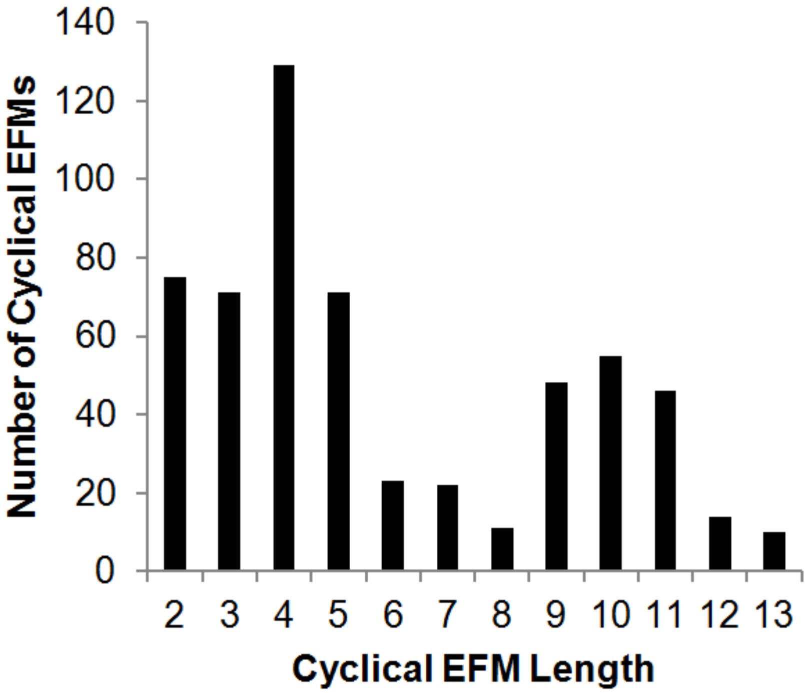


Figure 3-5: Cyclical EFM Distribution for module 141811 (Figure 3-3F)

3.4.4 Many Different Cofactors and Hub Metabolites are Involved in Substrate Cycles

An important motivation of this work was to highlight that many substrate cycles exist that consume/generate cofactors other than ATP. Table 2 enumerates the number of unique cyclical EFMs among all modules that have a net production and consumption of selected cofactors or hub metabolites. Cytosolic hydronium, phosphate, and sulfate ions have the most number of cyclical EFMs with a net consumption or production, reflecting the number of unique substrate cycles involved in membrane transport (Module 143830, Figure 3-3B). We found that the number of cyclical EFMs that use NADH(c)/NAD(c) is similar to the number of cyclical EFMs that use NADPH(c)/NADP(c). Many reversible reactions are catalyzed by either cofactor, and thus our algorithm identifies cycles where one reaction consumes NADH(c) in one direction and the other reaction consumes NADPH(c) in the reverse direction. Lastly, we find that many cyclical EFMs utilized deoxy-nucleotide triphosphates, suggesting substrate cycles may be involved in nucleotide metabolism.

Table 2: List of cofactors and the number of cyclical EFMs producing and consuming them.

Hub Metabolite	Number of Cyclical EFMs: Net Consumed	Number of Cyclical EFMs: Net Produced
H+(PG)(c)	70504	70446
Pi(c)	55051	55102
Sulfate(c)	33046	33046
H2O(c)	15015	14960
ATP(c)	155	217
ADP(c)	208	152
NADH(c)	24	24
NADPH(c)	19	20
UDP(c)	106	151
AMP(c)	0	21
GTP(c)	69	45
CTP(c)	94	47
dATP(c)	143	120
dGTP(c)	58	58
dAMP(c)	27	37
dUTP(c)	155	96

3.5 Discussion

In this study, we highlight that computational tools can greatly augment the discovery of novel substrate cycles in metabolic networks that would be otherwise difficult by simply surveying a two-dimensional map of textbook metabolic pathways. For example, for module 143829 (Figure 3-3C, Table 1), we list a cyclical EFM involving propionate metabolism. However, this potential substrate cycle would not have been identified by solely glancing at the interactive online KEGG pathway map for propionate metabolism because the metabolite 2-Methyl-3-oxopropanoate at the completion of this work is not classified into a pre-assigned pathway. As such, computational motif searches on complete metabolic networks is advantageous in that all reactions that take place in a specific cell type or organism are a part of the search, as opposed to just those displayed in a map representation. However, exhaustive motif searches on large scale networks are only computationally feasible for motifs comprised of a small number of nodes (Grochow and Kellis, 2007). For example, identifying all directed cycles spanning just two reaction nodes is feasible for a large scale metabolic network comprising a few thousand reactions, but enumeration of all directed cycles spanning multiple reaction steps is computationally inefficient with existing algorithms, because the number of cycles may grow exponentially with the number of vertices.

Gebauer and coworkers recently described the importance of computationally identifying substrate cycles that span multiple reaction steps by searching for cyclical EFMs. Unfortunately, due to the current computing limitations for exhaustive EFM enumeration on large scale networks, any approach to tackle this problem is inherently limited to identifying a targeted subset of all possible substrate cycles in the network. For example, the same group utilized the

EFMevolver (Grosse *et al.*, 2009) algorithm, which targets a specific subset of cyclical EFMs that produce a certain metabolite, in this case ATP (an artificial ADP phosphorylation reaction was added to be coupled to these substrate cycles) (Gebauer *et al.*, 2012). In our work, we relax the constraint of targeting only ATP consuming cyclical EFMs, and instead utilize a constraint on the search space to a systematically determined local neighborhood of reactions, or modules. In this manner, we compromise our ability to find longer cyclical EFMs, as evidenced by the distribution of cyclical EFM lengths (Figure 3-5) obtained for the module with the largest number of reactions for which EFMtool completed. This distribution spans between 2 and 13 reaction steps, compared to lengths of up to 100 reaction steps identified by Gebauer et al. (Gebauer *et al.*, 2012). While our approach likely misses many longer substrate cycles, we can identify substrate cycles that utilize many different cofactors other than just ATP, as our EFM computation on the modules was exhaustive. Determining local neighborhoods for motif searches can be challenging, and often invokes arbitrary rules such as defining a fixed radius (path distance) of search around a reaction of intereset. In this regard, ShReD-based modules provide a systematically derived localization of reaction groups enriched with cyclical interactions, making them conducive to capturing substrate cycles in the context of their metabolic function.

Thus far, we have been meticulous about not using the terms ‘cyclical EFM’ and ‘substrate cycle’ interchangeably, because the identified cyclical EFMs are solely based on the stoichiometry and specified reaction directionality in the model. That is, they only represent the possibility of substrate cycling for a set of reactions and do not necessarily determine whether or not the substrate cycle is actually active *in vivo*. For example, many identified cyclical EFMs involve two reversible reactions where one consumes NADP and produces NADPH in the forward direction, and the other consumes NADH and produces NAD in the reverse direction (Table 1, Module 144976, Figure 3-3A). However, whether or not these reactions can indeed operate in opposite directions at the same time with a net conversion of NADP to NADPH at the expense of NADH (or vice versa) depends on the intracellular concentrations of the species involved and the ΔG driving force. In one such example, Sazanov and coworkers report substrate cycling between isocitrate and ketoglutarate, where NAD-dependent isocitrate dehydrogenase (IDH) catalyzes the conversion of isocitrate to ketoglutarate and NADP-dependent IDH catalyzes the conversion of ketoglutarate back to isocitrate (Sazanov and Jackson, 1994).

The presented methodology for the identification of potential substrate cycles obviously hinges on the accuracy of the model used. Metabolic models are constantly updated with increased knowledge as to which genes are expressed in specific organisms or human tissue. Inaccurate model curation will inevitably impact the results of the presented algorithm. For example, at the completion of this work, the reaction catalyzed by CMP-Neu5Ac hydroxylase for N-Glycolylneuraminate (NeuNGc) synthesis is still present in the hepatonet1 model (Table 1, Module 143829). Our algorithm identifies a substrate cycle incorporating this reaction for a net production or consumption of NADH(c) depending on the direction of cycling. However, reports suggest that the gene for CMP-Neu5Ac hydroxylase is absent in human tissue (Chou *et al.*, 2002). Confusingly, this reference was cited by the authors of the hepatonet1 model as justification for including this reaction. The onus is therefore on the user of the algorithm to decide if the identified cyclical EFMs represent biologically feasible substrate cycles in the particular tissue or cell type.

This methodology also relies on the accuracy of the modularization of the network, and ShReD-based partitioning is a reaction assignment algorithm that is not always perfect. If the systematically uncovered modularity of the network is sub-optimal, then it is possible that some recognizable substrate cycles in literature would not be uncovered by the algorithm because the reactions involving them may split early with the ShReD-based partition algorithm, and the modules for which those reactions are present together, close to the parent module in the hierarchy, may have too many reactions for EFMtool to complete. For example, Peterson and coworkers have previously reported cycling in the rat liver involving pyruvate carboxylase, malate dehydrogenase, and malic enzyme with a net loss of NADH and NADPH (Petersen *et al.*, 1995). However, while all the reactions required for this substrate cycle are present in the human hepatonet1 model, our algorithm does not identify this substrate cycle because cytosolic malic enzyme only remains together with mitochondrial pyruvate carboxylase and malate dehydrogenase in 141802, which contained too many reactions for EMFtool to complete. As such, the modularity of the network itself may need improvement. For example, in this study, all edges in the reaction-centric graph were treated as static and weighted equally for the ShReD distance metric. However, we show in the next chapter that metabolic-flux weighted networks will result in different modularity, depending on the metabolic state of the system. If the aforementioned substrate cycle was highly active with a relatively large cyclic flux, then a network modularity based on metabolic flux data may reflect this tight engagement between reactions by placing those reactions in the same module (Sridharan *et al.*, 2012). However, high resolution flux data using isotope labeling are a daunting challenge for such a large scale network, and approximate flux data using Flux Balance Analysis can be used instead.

Prospectively, the discovery of new substrate cycles in metabolic networks using computational tools may yield novel drug targets in the context of obesity treatment or even cancer. For example, thiazolidinediones (TZDs) activate (PPAR)-ϒ in adipocytes, which induces glycerol kinase gene expression and promotes a futile cycle between triglyceride degradation and synthesis from fatty acids and glycerol (Guan *et al.*, 2002). They showed that TZD treatment of adipocytes reduces the secretion of free fatty acids, whose levels in serum are associated with obesity and insulin insensitivity. Moreover, the activity of this substrate cycle appears to be greatly enhanced in a murine animal model for cancer cachexia (Beck and M J Tisdale, 2004), suggesting that many substrate cycles are differentially expressed in tumor cells. In fact, the Cori cycle is another substrate cycle with increased activity in cachetic patients, where the lactic acid produced in the tumor is converted back to glucose in the liver at the expense of six ATP molecules (Michael J Tisdale, 2005). In addition, the cyclical EFM we report in Module 142887 (Figure 3-3E, Table 1) is involved in folate metabolism, and folate receptors are up-regulated in cancer cells, providing targets for cancer therapy (Low *et al.*, 2008). However, the role of this potential substrate cycle in cancer metabolism remains to be investigated. While the hepatonet1 model lists the reaction catalyzed by dihydrofolate reductase as reversible, most report that the reaction favors the direction of making tetrahydrofolate at the expense of NADPH (Bailey and Ayling, 2009). Indeed, many inter-organ substrate cycles, such as the Cori cycle, would not be computationally identified in tissue-specific metabolic models, which may prompt one to investigate metabolic networks spanning several organs in future work.

3.6 Conclusion

In this chapter, we used ShReD-based modules to focus the search of cyclical EFMs to within modules. In this manner, the EFM calculation was exhaustive for all but a few modules. Potential substrate cycles were identified that had a net production and consumption of a wide range of cofactors and hub metabolites. However, we have shown that this methodology hinges on the accuracy of the modularization, in that some substrate cycles may not be identified because reactions split early in the hierarchical partitioning. We therefore seek ways to improve ShReD-based modularity algorithm. It was mentioned that weighting edges based on metabolic flux data can allow the partitioning of metabolic state-dependent networks. The next chapter delves into the details of this novel weighting scheme.

4 Metabolic Flux-Based Modularity Using ShReD

4.1 Abstract

Graph-based modularity analysis has emerged as an important tool to study the functional organization of biological networks. However, few methods are available to study state-dependent changes in network modularity using biological activity data. We develop a weighting scheme, based on metabolic flux data, to adjust the interaction distances in a reaction-centric graph model of a metabolic network. The weighting scheme was combined with a hierarchical module assignment algorithm featuring the preservation of metabolic cycles to examine the effects of cellular differentiation and enzyme inhibitions on the functional organization of adipocyte metabolism. Our analysis found that the differences between various metabolic states primarily involved the assignment of two specific reactions in fatty acid synthesis and glycerogenesis. Our analysis also identified cyclical interactions between reactions that are robust with respect to metabolic state, suggesting possible co-regulation. Comparisons based on cyclical interaction distances between reaction pairs suggest that the modular organization of adipocyte metabolism is stable with respect to the inhibition of an enzyme, whereas a major physiological change such as cellular differentiation leads to a more substantial reorganization.

4.2 Introduction

The topology of interactions in a biological network is often studied by modeling the network as a graph, which allows the use of established algorithms and metrics such as shortest path analysis (Floyd, 1962) and betweenness centrality (Girvan and M. E. J. Newman, 2002). Graph theoretical models have yielded useful insights into not only the global topology of biological networks, but also local interactions that form distinct substructures, frequently referred to as modules (Holme *et al.*, 2003; J. Zhao *et al.*, 2006). Indeed, there is growing consensus that many types of biological networks possess modular character. Hierarchically arranged modules have been identified in metabolic networks, where larger, more heterogeneous subnetworks comprise smaller, more cohesive subnetworks (Papin *et al.*, 2004; Ravasz *et al.*, 2002). Hierarchical modularity has also been observed for gene interaction networks (Treviño *et al.*, 2012) and protein interaction networks (Yook *et al.*, 2004).

Despite the important insights obtained from topological analysis, almost all of the graph-based studies to date have examined a biological network under a single static condition (Ideker and Krogan, 2012). For instance, Potapov and coworkers note that shortest path analysis, applied to a static network, may offer limited information because the length of an edge in the graph model may not correlate well with the overall efficiency of a particular biochemical transformation represented by the edge (Potapov *et al.*, 2008). There is increasing evidence that biological network organization is dynamic and state dependent, which cannot be adequately studied from a static point of view. As a result, there has been growing interest in augmenting the topological information of biological networks for graph-based analysis with observed activity data. Recently, Tang and coworkers used gene expression data to construct time-course protein interaction networks, and found that functional modules detected in the time-course networks more closely matched known regulatory complexes than those detected in the static networks (X. Tang *et al.*, 2011). In another example, Greenblum and coworkers constructed a metagenomic network of the human gut microbiome using gene expression data, and showed that state-specific networks representing lean or obese individuals exhibited different topological properties, including modularity (Greenblum *et al.*, 2011). Similarly, Taylor and coworkers found that dynamic changes in the organization of the protein-protein interaction network, rather than expression levels of individual proteins, correlated strongly with breast cancer prognosis (Taylor *et al.*, 2009). Interestingly, mutations in hub proteins connecting different modules were found to be more frequently associated with cancer phenotypes than mutations in hub proteins that are highly connected with other proteins in the same modules, suggesting that alterations in global modularity may occur in cancer.

In the case of a metabolic reaction network, gene or even protein expression data may not best capture the interactions between the network's components, as mRNA levels or enzyme concentrations do not necessarily correlate with reaction rate or metabolite turnover. A more comprehensive snapshot of the physiological state may be provided by a metabolic network's reaction flux distribution, which directly reflects the relative engagements of enzymes, integrating the various layers of regulatory processes active in the cell. Intuitively, the flux of a reaction can be used to weight the interaction mediated by the reaction. For example, Yoon and coworkers applied flux-based weights to adjust the edge distances in a graph model of murine adipocyte metabolism, and thereby reflect metabolic state-dependent variations in the interactions between metabolite pools (Si *et al.*, 2007; Yoon *et al.*, 2007). While intuitive, this weighting scheme assumes that the metabolic network is modeled as a metabolite centric graph, where the edges represent reactions. For the purpose of studying the interactions between enzymes, it is often useful to model the metabolic network as a reaction centric graph, where the nodes represent enzymes and edges represent interactions between the enzymes mediated by metabolite substrates and effectors (Ma *et al.*, 2004). The benefit of a reaction-centric graph, particularly in the context of modularity analysis, is that a metabolite is not constrained to a module. Instead, a metabolite is more appropriately modeled as a shared resource, and reactions define the functional identity of a module. To our knowledge, a scheme to weight the edges of a reaction-centric graph has not yet been described in the literature. The purpose of this study was therefore to develop a generally applicable method for incorporating activity data such as metabolic flux into modularity analysis using graph models where the nodes, rather than the edges, represent the network's functional components.

In Chapter 2, we defined a new metric, termed Shortest Retroactive Distance (ShReD), to capture feedback and other cyclical interactions in a metabolic network (Sridharan *et al.*, 2011). Based on the earlier work of Saez-Rodriguez and coworkers on retroactivity (Saez-Rodriguez *et al.*, 2008), ShReD was used to solve for modular partitions that would minimize cyclical interactions between modules while maximizing such interactions within a module. While the earlier work on retroactivity focused on nearest neighbor interactions, for example mediated by the product of a reversible reaction, the ShReD-based analysis also considered interactions between distant parts of a network. In the present study, we further expand the use of ShReD as a modularity analysis metric by developing a weighting scheme to reflect phenotypic state-dependent variations in reaction-to-reaction interactions. We focus on flux data due to the integral nature of the information content in such data, reflecting the functional outcomes of transcriptional, translational, and post-translational mechanisms of enzyme activity regulation. Flux data can be obtained using a number of different methods, including isotopic (typically ^{13}C) labeling, metabolic flux analysis (MFA), and flux balance analysis (FBA), Generally, mathematical model-based analysis of isotopic enrichment of multiple metabolite pools offers the greatest resolution. Flux balance analysis is a constrained optimization based approach typically used to estimate fluxes in conjunction with a metabolic objective function. The problem is usually severely underdetermined in FBA. In the present chapter, we used a constrained optimization based approach to estimate metabolic fluxes, but without assuming a metabolic objective. Rather, we minimized the sum of squared differences between the measured and estimated exchange fluxes, as the problems were well constrained. Applied to a model of adipocyte metabolism, ShReD-based modules obtained using flux weights more consistently reflected recognizable functions of established pathways compared to the modules obtained without the weights. Comparisons of modules obtained using several different flux sets representing distinct metabolic states identified robust reaction pairs that repeatedly partitioned into the same module across many levels of modular hierarchy, suggesting possible co-regulation.

4.3 Methods

4.3.1 Adipocyte Model and Fluxes

A stoichiometric network model of adipocyte central carbon metabolism was formulated by slightly modifying a previously published model (Si *et al.*, 2007). The modifications were as follows. Reactions were removed for ketone body metabolism, because these reactions carried negligible flux. Reactions were added for glyceroneogenesis to allow the synthesis of glycerone-phosphate from phosophoenolpyruvate. The number of reactions and metabolites in the modified model were 72 and 79, respectively, with 48 independent steady state balances and 22 measured exchange rates. The system was underdetermined by a degree of two. Metabolic flux distributions were calculated for four different phenotypic states: immature adipocyte (day 4 post-induction), mature adipocyte (day 12 post-induction), mature adipocyte treated with an inhibitor for lactate dehydrogenase (LDH), and mature adipocyte treated with an inhibitor for pyruvate carboxylase (PCX). Rates of metabolite uptake and output (exchange rates) describing these phenotypic states were taken from our previous work (Si *et al.*, 2007, 2009). Fluxes were calculated by minimizing the sum of squared differences between measured and calculated metabolite exchange rates subject to stoichiometric balance constraints. The reaction definitions of the adipocyte model and flux distributions corresponding to the four phenotypic states are listed in Appendix A.

4.3.2 Flux-based ShReD

In the previous chapter, we defined the ShReD metric to characterize the connectivity between two biochemical network components that interact retroactively (Sridharan *et al.*, 2011). Connectivity relationships between reactions in a metabolic network as defined by stoichiometry can be modeled using a directed graph with vertices representing reactions and edges indicating a directional interaction between connected reactions. Edges are drawn between two reactions if the product of one reaction is a reactant of the other reaction. Based on this graph model, the ShReD of reaction nodes i and j is computed as the sum of the shortest path distance from node i to j and the shortest return path from node j to i. In computing the shortest path distance and shortest return path distance, each edge can be assigned the same unit distance to consider a nominal state where the interactions between reactions are solely determined by network topology. Alternatively, each edge can be assigned a different weight that reflects the engagement of the biochemical interaction represented by the edge. For a pair of metabolic reactions, a quantitative measure of this interaction can be obtained from the flux of the intermediary metabolite. Consider the example in Figure 4-1A.

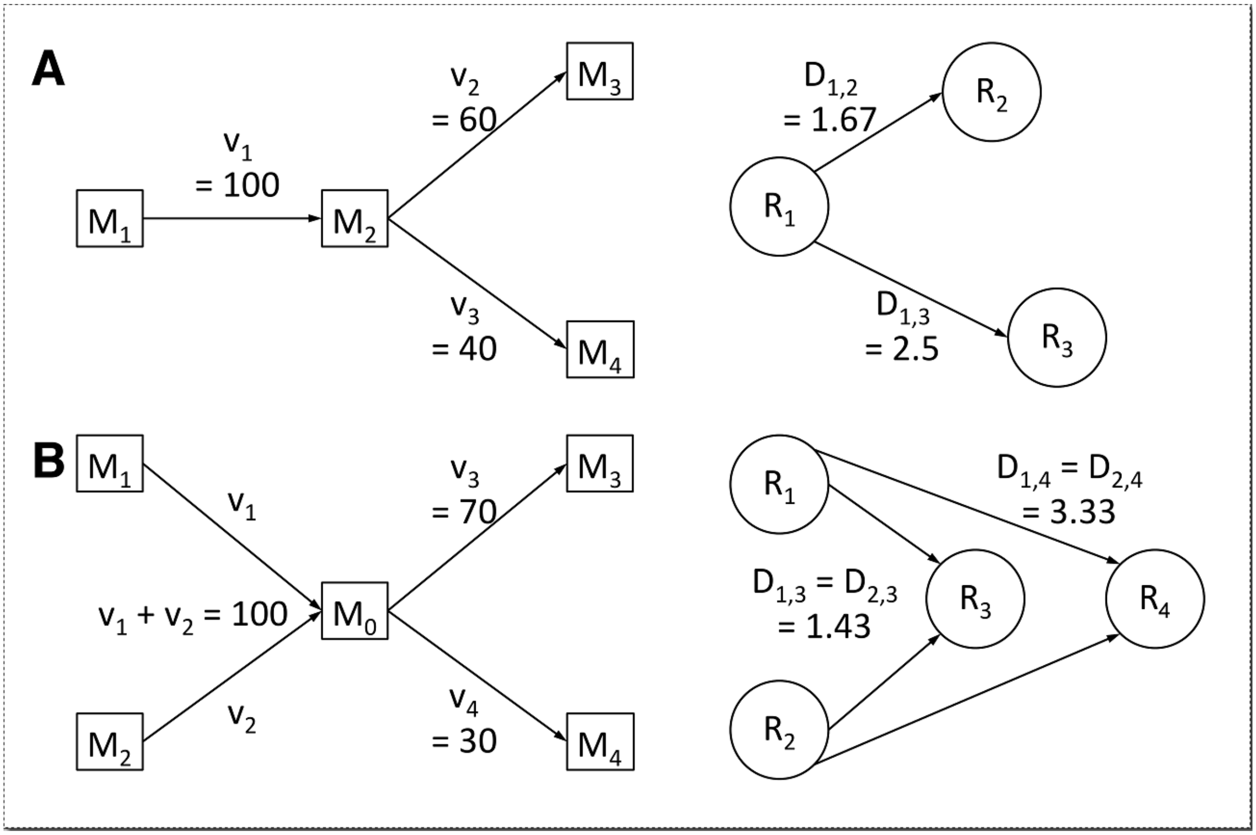


Figure 4-1: Metabolic Flux-based Edge Weighting

Examples of metabolic flux-based edge weighting in the case of (A) one reaction producing the intermediary metabolite and (B) multiple reactions producing the intermediary. In panel (A), v_i refers to the flux of reaction R_i.

Reaction R_1 produces 100 mol/min of metabolite M_2, of which 60 mol/min is directed towards R_2 and the remainder towards R_3. Assuming that the pool of M_2 is homogeneous, we attribute a stronger influence of R_1 on R_2 relative to R_3. Intuitively, a stronger influence is modeled as a smaller edge weight (shorter path distance), whereas a weaker influence is modeled as a larger edge weight (longer path distance). Formally, we define the edge distance between a connected pair of reaction nodes as the *inverse of the fraction of the intermediate metabolite production flux that is directed towards the destination reaction node*. In the example of Figure 4-1A, the dimensionless edge distance $D_{1,2}$ between R_1 and R_2 is given by:

$$D_{1,2} = 1 \Big/ \frac{60}{100} = 1.67 \tag{4.1}$$

The edge distance between R_1 and R_3 is given by:

$$D_{1,3} = 1 \Big/ \frac{40}{100} = 2.50 \tag{4.2}$$

The above definition can be extended in a straightforward manner to cases when more than one reaction produces the metabolite that mediates the interaction between two reactions (Figure 4-1B). These cases often involve energy cofactors such as ATP, NADH, or NADPH, which are produced and consumed by many reactions. In the example of Figure 4-1B, the total steady state flux of M_0 is 100 mol/min, of which 70 and 30 mol/min is directed towards R_3 and R_4, respectively. Based on the assumption that the metabolite pool is homogeneous and the contributions of the upstream reactions to this pool are indistinguishable, the directed interaction from R_1 to R_3 is the same as the interaction from R_2 to R_3. The weighting would be the same if v_1 and v_2 are each 50, or if v_1 is several orders of magnitude smaller than v_2. Even if $v_1 = 0.01$, it has to be assumed that 70% of that small flux is directed towards R_3, because the source of the intermediary metabolite flux cannot be distinguished by the downstream enzymes. Similarly, the interaction from R_1 and R_4 is the same as the interaction from R_2 to R_4. Generalizing for a pair of reactions R_i and R_j connected through an intermediary metabolite M produced by an arbitrary number reactions N, the edge distance from node R_i to R_j is given by:

$$D_{ij} = \sum_{k=1}^{N} v_k \Big/ v_j \tag{4.3}$$

In equation 3, the index k refers to the set of reactions R_k that produce the intermediary metabolite M, v_k is the flux of R_k, and v_j is the flux of the reaction R_j. When v_j is close to zero, the corresponding edge distance is very large, as is any ShReD that includes this edge. In such cases, allowing a ShReD to reach an arbitrarily large value could exaggerate the numerical difference between reactions whose fluxes are not statistically different from zero. Therefore, the value of a flux-weighted ShReD was capped with an upper bound. For numerical convenience, the cap was set at 100, as fewer than 5 % of all ShReDs calculated in this study exceeded this value. The calculation of ShReDs based on flux weights is illustrated in Figure 4-2. Distinct flux distributions (Figures 4-2A and 4-2C) can result in different ShReDs for the same reaction pair (Figures 4-2B and 4-2D).

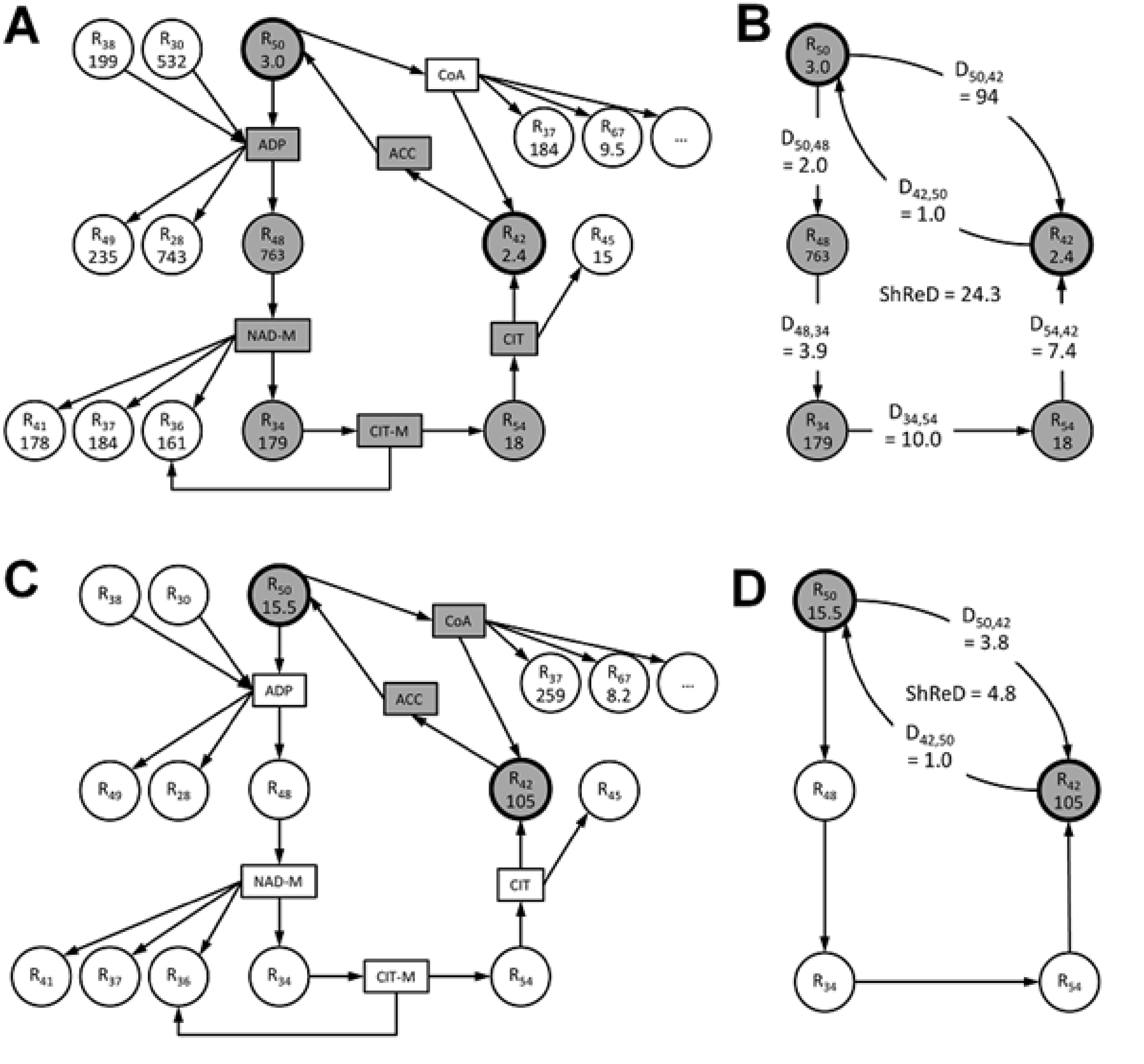


Figure 4-2: Example of Metabolic Flux-based ShReD

Circles and boxes show reactions and metabolites, respectively. The top and bottom numbers in the circles refer to the reaction number and corresponding flux, respectively. The ShReDs are highlighted with grey circles and boxes. (A) Day 4 model flux distribution around reaction nodes comprising the ShReD for the reaction pair [R42, R50]. The reactions comprising the ShReD are as follows. R_{50}: palmitate synthesis, R_{48}: oxidative phosphorylation, R_{34}: pyruvate dehydrogenase and citrate synthase, R_{54}: mitochondrial transport of 2-oxoglutarate and malate, R_{42}: citrate lyase. (B) Reaction-to-reaction distances calculated using the fluxes shown in panel (A) and equation 3. Note that the path from R_{50} to R_{42} in $\text{ShReD}_{42,50}$ proceeds through R_{48}, R_{34}, and R_{54} due to the long weighted distance from R_{50} to R_{42}, which in turn is due to the relatively small contribution of R_{50} to the CoA flux. (C) Day 12 model flux distribution around reaction nodes comprising $\text{ShReD}_{42,50}$. (D) Reaction-to-reaction distances calculated using the fluxes shown in panel (A) and equation 3. Note that $\text{ShReD}_{42,50}$ for the Day 12 model involves only R_{42} and R_{50}, as the contribution of R_{50} to the CoA flux is significantly larger compared to the Day 4 model.

4.3.3 Partition Algorithm

Partitions of flux-weighted and unweighted network models were generated using Newman's community detection algorithm (M. E. J. Newman, 2006) similar to what was described in the previous chapter. The overall algorithm flow is shown in Figure 4-3

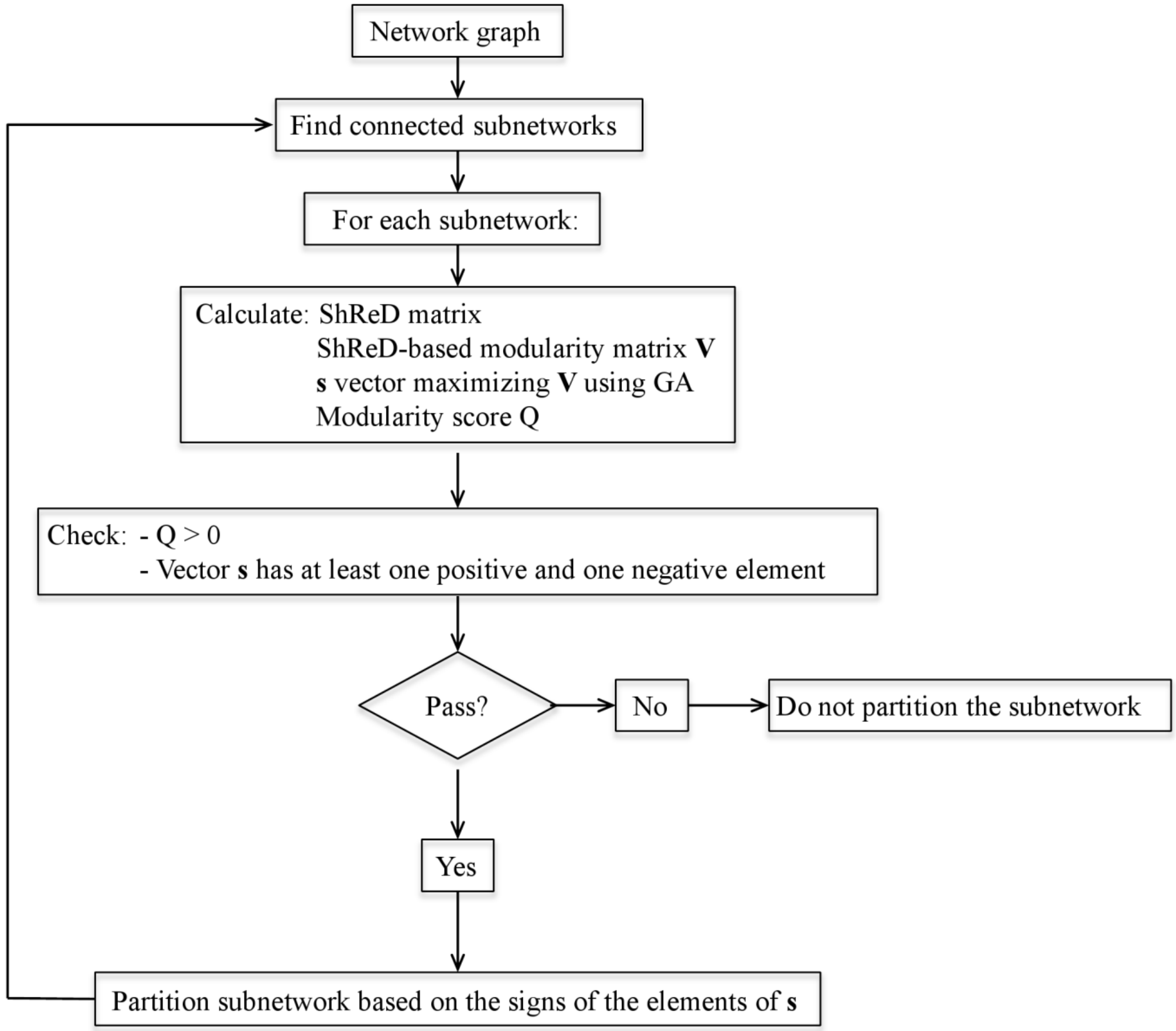

Network graph
Find connected subnetworks
For each subnetwork:
Calculate: ShReD matrix
ShReD-based modularity matrix **V**
s vector maximizing **V** using GA
Modularity score Q
Check: - Q > 0
- Vector **s** has at least one positive and one negative element
Pass?
No
Do not partition the subnetwork
Yes
Partition subnetwork based on the signs of the elements of **s**

Figure 4-3: Algorithm workflow for ShReD-based partitioning using flux-weighted edges.

.

Briefly, the partitioning algorithm begins by finding the connected subnetworks in the parent network using a breadth-first traversal algorithm (Cormen *et al.*, 2001), as it is possible that the parent network, represented as a reaction centric graph, may not be fully connected. Each connected subnetwork is then partitioned into two daughter subnetworks to maximize a modularity score. Applied recursively, the algorithm produces a hierarchical tree of modules. Unlike our previous work, we do not require each daughter subnetwork to contain at least one cycle as a criterion for partition. It is sufficient that at least one daughter subnetwork contains at least one cycle. This relaxation allows the algorithm to find solutions (reaction node assignments) that result in a partition where single reaction nodes peel off from a larger subnetwork. While the single reaction nodes obviously cannot possess a cycle, this should not preclude further partitioning of the larger subnetwork.

4.3.4 Modularity Matrix

In the previous chapter, we computed a modularity score based on the difference between the actual and expected ShReD assuming that each edge in the graph model has equal length, or weight. In this study, we modify the modularity matrix from which the modularity score is calculated to account for the skewness of weighted ShReD distributions. The desired modularity matrix **V** has a positive entry V_{ij} if the corresponding ShReD between a pair of reaction nodes is small relative to the expected ShReD, whereas it has a negative entry if the corresponding ShReD is large. Due to the skewing effect of the flux weights on the ShReD distribution, the determination of whether a weighted ShReD is small or large relative to expectation was based on a log ratio. Formally, we define an entry V_{ij} in the modularity matrix **V** as follows.

$$V_{ij} = \ln\left(\frac{p_{ij}}{1-p_{ij}}\right) \qquad (4.4)$$

In equation 4, p_{ij} is the fraction of all weighted ShReDs involving reaction R_i or R_j that is longer than the ShReD between R_i and R_j ($ShReD_{ij}$). If exactly half of all ShReDs involving R_i or R_j are longer (or shorter) than $ShReD_{ij}$, then V_{ij} is zero. Otherwise, V_{ij} is positive or negative depending on the rank of $ShReD_{ij}$ relative to all other ShReDs involving R_i or R_j. As an example, consider the subnetwork shown in Figure 4-9B (module #7249). The flux weighted ShReD matrix for this subnetwork is shown in Figure 4-4.

$$ShReD =$$

$$\left[\begin{array}{lcccccccccccccc}
 & \mathbf{R_{24}} & \mathbf{R_{25}} & \mathbf{R_{26}} & \mathbf{R_{27}} & \mathbf{R_{28}} & \mathbf{R_{29}} & \mathbf{R_{30}} & \mathbf{R_{31}} & \mathbf{R_{33}} & \mathbf{R_{35}} & \mathbf{R_{47}} & \mathbf{R_{51}} & \mathbf{R_{52}} & \mathbf{R_{53}} \\
\mathbf{R_{24}} & 0 & 9.82 & 9.82 & 10.8 & 9.82 & 12.4 & 12.4 & 15.8 & 100 & 22.7 & 12.1 & 57.7 & 60.3 & 22.7 \\
\mathbf{R_{25}} & 9.82 & 0 & 9.82 & 10.8 & 9.82 & 12.4 & 12.4 & 15.8 & 100 & 22.7 & 12.1 & 57.7 & 60.3 & 22.7 \\
\mathbf{R_{26}} & 9.82 & 9.82 & 0 & 8.56 & 7.54 & 10.1 & 10.1 & 13.5 & 100 & 20.4 & 9.88 & 55.4 & 58.0 & 20.4 \\
\mathbf{R_{27}} & 10.8 & 10.8 & 8.57 & 0 & 3.35 & 5.77 & 6.91 & 9.36 & 100 & 16.2 & 3.35 & 53.6 & 56.1 & 16.2 \\
\mathbf{R_{28}} & 9.82 & 9.82 & 7.54 & 3.35 & 0 & 4.75 & 5.89 & 6.00 & 100 & 12.9 & 2.33 & 50.2 & 52.8 & 12.9 \\
\mathbf{R_{29}} & 12.4 & 12.4 & 10.1 & 5.77 & 4.75 & 0 & 5.89 & 7.14 & 100 & 12.9 & 4.75 & 52.6 & 55.2 & 12.9 \\
\mathbf{R_{30}} & 12.4 & 12.4 & 10.1 & 6.91 & 5.89 & 5.89 & 0 & 7.14 & 100 & 12.7 & 5.89 & 53.7 & 56.3 & 12.7 \\
\mathbf{R_{31}} & 15.8 & 15.8 & 13.5 & 9.36 & 6.00 & 7.14 & 7.14 & 0 & 100 & 18.9 & 8.33 & 56.2 & 58.8 & 18.9 \\
\mathbf{R_{33}} & 100 & 100 & 100 & 100 & 100 & 100 & 100 & 100 & 0 & 100 & 100 & 100 & 100 & 100 \\
\mathbf{R_{35}} & 22.7 & 22.7 & 20.4 & 16.2 & 12.9 & 12.9 & 12.7 & 18.9 & 100 & 0 & 15.2 & 63.1 & 65.7 & 12.7 \\
\mathbf{R_{47}} & 12.1 & 12.1 & 9.88 & 3.35 & 2.33 & 4.75 & 5.89 & 8.33 & 100 & 15.2 & 0 & 50.2 & 52.8 & 15.2 \\
\mathbf{R_{51}} & 57.7 & 57.7 & 55.4 & 53.5 & 50.2 & 52.6 & 53.8 & 56.2 & 100 & 63.1 & 50.2 & 0 & 2.59 & 63.1 \\
\mathbf{R_{52}} & 60.3 & 60.3 & 58.0 & 56.1 & 52.8 & 55.2 & 56.3 & 58.8 & 100 & 65.7 & 52.7 & 2.59 & 0 & 65.7 \\
\mathbf{R_{53}} & 22.7 & 22.7 & 20.4 & 16.2 & 12.9 & 12.9 & 12.7 & 18.9 & 100 & 12.7 & 15.2 & 63.1 & 65.7 & 0
\end{array}\right]$$

Figure 4-4: ShReD matrix of module #7249 in the Day 12 partition.

There are a total of 26 ShReDs involving R24 or R31, including the ShReD between R24 and R31 ($ShReD_{24,31}$). Of these, $ShReD_{24,31}$ ranks 11th in terms of length. Applying equation 4, $p_{24,31} = 10/25 = 0.4$, and $V_{24,31} = -0.41$. If $p_{ij} = 0$, p_{ij} is arbitrarily set to 0.01. The smallest V_{ij} value is thus -4.60, which is on the same order of magnitude as the other entries in the modularity matrix.

4.3.5 Optimization of the Modularity Score

To generate a partition, we assign each reaction in a subnetwork into one of two daughter subnetworks as in the previous chapter. The goal is to find a set of assignments, represented by a binary vector **s**, that maximizes the modularity score (M. E. J. Newman, 2006). The modularity score Q is defined based on the modularity matrix **V**:

$$Q = \sum_{i}\sum_{j} V_{ij} s_i s_j = \mathbf{sVs}^T \qquad (4.5)$$

Each element s_i or s_j of vector **s** has a value of either -1 or 1. An increase in Q is obtained in two cases: if V_{ij} is positive and reactions i and j are assigned to the same subnetwork ($s_i = s_j = 1$ or $s_i = s_j = -1$), or if V_{ij} is negative and the two reactions are assigned to different subnetworks ($s_i = 1$ and $s_j = -1$ or vice versa). A solution to the maximization problem can be found using a number of different optimization methods. For example, an approximate solution can be obtained using eigenvalue decomposition (M. E. J. Newman, 2006). In this study, we used a genetic algorithm (GA). While the GA was computationally less efficient than the eigenvalue decomposition method, it yielded superior solutions (**s** vectors) with larger Q scores. The GA was implemented using custom code written in MATLAB with the following parameters. The initial population of solutions comprised 100 randomly generated **s** vectors. The population size was kept constant. A fixed fraction (60 %) of the solutions was selected for reproduction based on fitness (Q score). New individuals were bred through crossover and mutation. During crossover, an element in the offspring **s** vector was assigned the same value as the corresponding elements in the parent **s** vectors if the values were the same in both parents. Otherwise, the element was randomly assigned either -1 or 1. The mutation (sign change) rate was set to 20 %. The GA terminated when the average Q score of the population reached a plateau with an absolute slope < 0.05 with respect to generation number. The fittest solution (**s** vector with the largest Q score) generated over the course of the GA was used for the partition. For the subnetworks encountered in this study, termination was reached generally within 200 generations. For the example subnetwork of Figure 4-9B (Module #7249), the GA terminated in 117 generations, and clearly outperformed the eigenvalue solution (Figure 4-5).

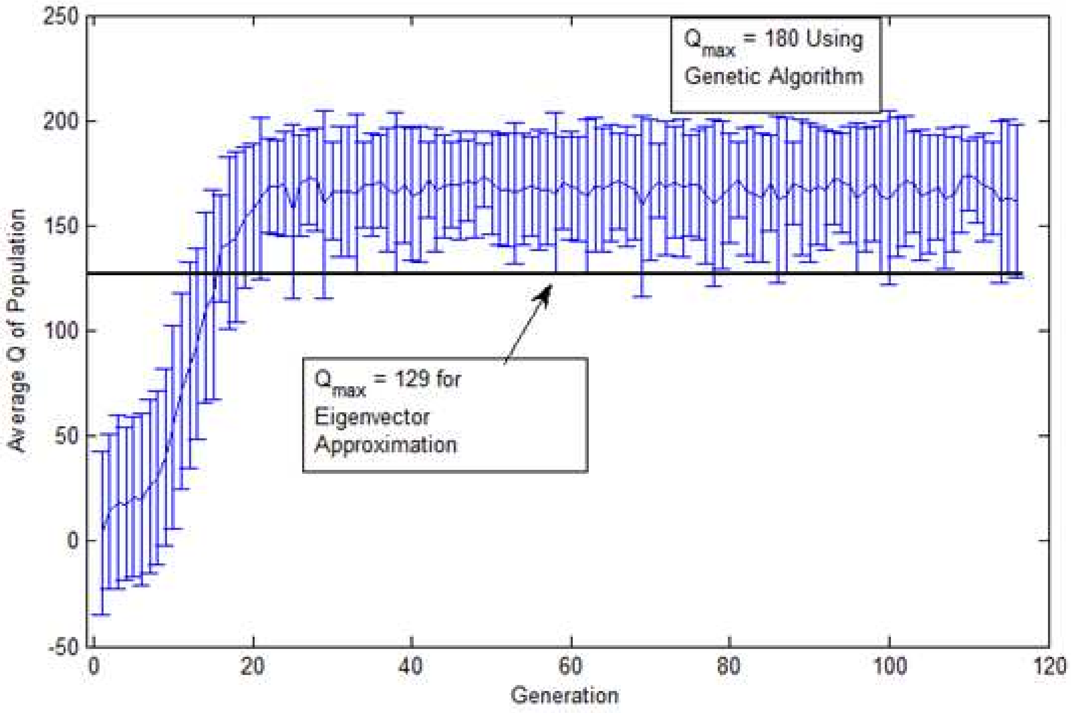


Figure 4-5: GA solution outperforms eigenvector approximation

In cases where the subnetwork size was sufficiently small (< 9 reactions), an exhaustive search was performed to find a globally optimal solution. The runtime for the complete partitioning of the Day 12 model was 180 seconds using the GA and 85 seconds using the eigenvalue approximation on a laptop computer with a 2.2 GHz CPU (Intel Core 2 Duo) and 4 GB of physical memory.

4.3.6 Hierarchical Tree of Modules

The partitioning results are reported in the form of a hierarchical tree annotated with several properties. Each module is represented as a pie chart, where the size of each slice is proportional to the fraction of reactions that belong to the corresponding, pre-assigned canonical (textbook) grouping. The homogeneity index of a module corresponds to the fraction occupied by the largest slice in the pie chart. The homogeneity index therefore ranges from 0 to 1, where a larger number indicates greater homogeneity in terms of composition based on the canonical group assignments. The black lines connecting the nodes in the hierarchical tree represent ShReD-based partitions, whereas the red lines represent the formation of components from partitions that include disconnected components. The depth of a module is determined as the number of black edges traversed from the root node to the module. The height of a module is determined as the largest possible number of black edges traversed from the module to a terminal leaf node.

4.3.7 Partition Score for Reaction Pairs

To determine the correlation between modularity and flux weighted ShReD-based partitioning, we define a partition score H for a pair of reaction nodes by scaling the number of shared modules in the partition tree with respect to the partition depth of the terminal modules for the reaction nodes:

$$H_{ij} = \frac{Shared - 1}{2}\left[\frac{1}{m_i} + \frac{1}{m_j}\right] \qquad (4.6)$$

where *Shared* is the number of modules in the partition hierarchy that include both reactions i and j, and m_i and m_j are, respectively, the maximal depth of reactions i and j. The numerical range of H is thus from 0 to 1. A value of zero indicates that the two reactions are immediately separated after the first partition operation, whereas a value close to one indicates that the two reactions remain together in the same module through many rounds of partition operations.

4.3.8 Reaction Pair H-V Space Euclidean Distance

To assess the impact of metabolic state and its corresponding flux distribution on the hierarchical partition of reaction modules, a Euclidean distance is computed for each reaction pair in the H-V (partition score – modularity score) coordinate space from its original location corresponding to the first metabolic state to its new location corresponding to the second state. All coordinates are normalized to the mean partition score and modularity score of the corresponding flux-weighted partition.

4.4 Results

4.4.1 Effects of Weighting Edges on ShReD Distribution

Weighting the edges that define the interactions between the reaction nodes substantially impacted the overall distribution of ShReDs. In the un-weighted case, when all edges have the same unit distance, the ShReDs are close to normally distributed, with a skewness of -0.048 (Figure 4-6).

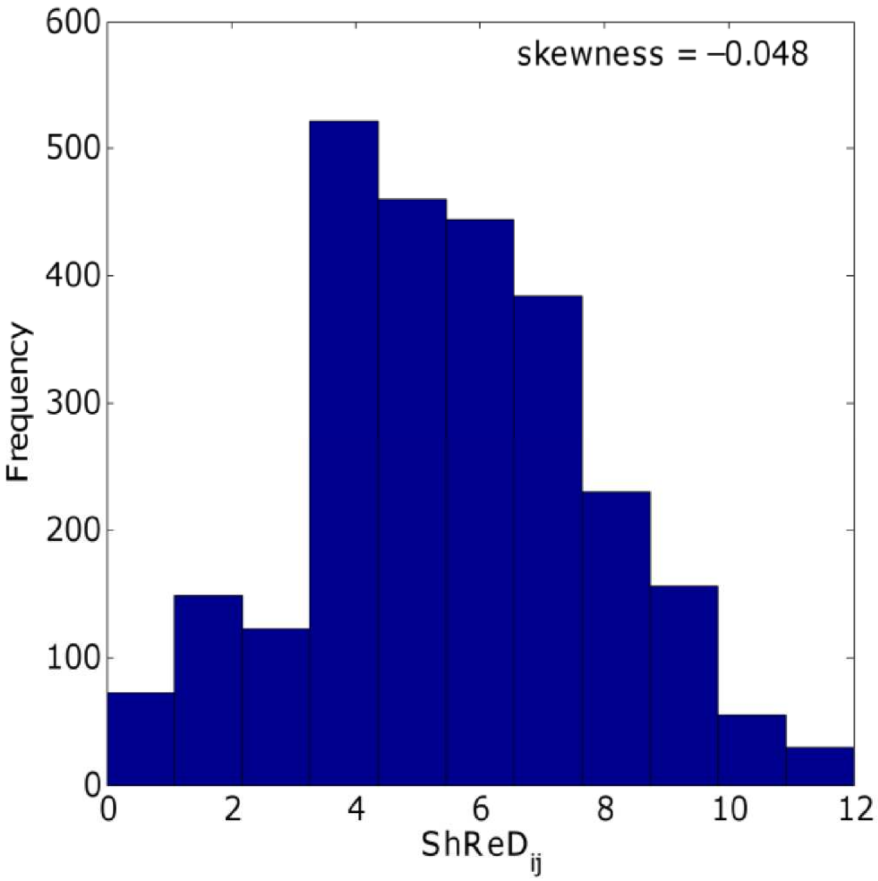


Figure 4-6: Distribution of ShReD values for the un-weighted adipocyte network

The histogram only shows ShReDs with value<100

In the weighted cases, the distributions clearly skew to the right. For example, the skewness of the Day 12 model is 1.57 (Figure 4-7).

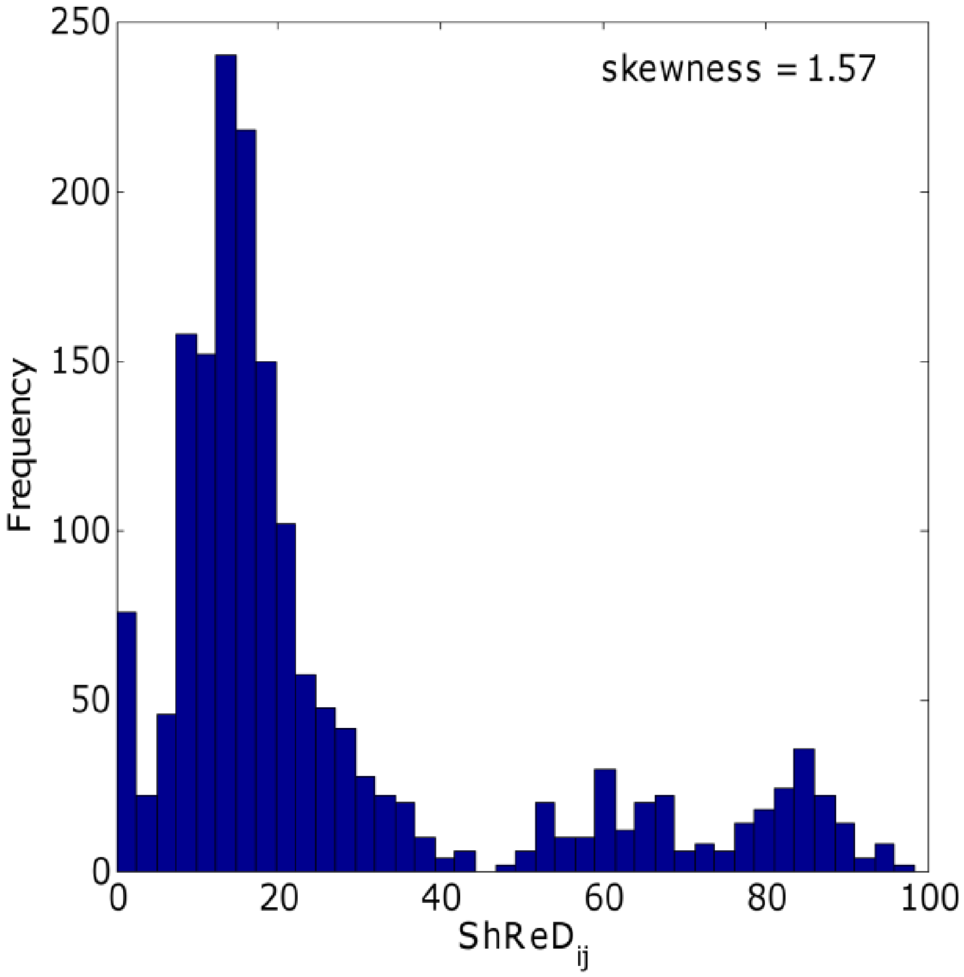


Figure 4-7: Distribution of ShReD values for Day 12 flux-weighted model

The histogram only shows ShReDs with value <100

This difference in ShReD distribution between the weighted and un-weighted cases motivated an adjustment from the modularity metric presented in the previous chapter. In the previous chapter, the entries for the modularity matrix were computed as the difference between the expected and actual ShReD of two reaction nodes, where the expected ShReD was calculated as the arithmetic average of all ShReDs involving either one of the two reaction nodes. This calculation assumed that the overall ShReD distribution is close to normal, and that the arithmetic average reasonably represents the expected ShReD between two nodes. With the incorporation of reaction flux-based weights, many reaction-to-reaction interactions were orders of magnitude weaker (and the corresponding graph distances were longer) than the average interaction, for example, due to the involvement of amino acid reactions whose fluxes were negligible compared to those of glucose and lipid metabolism. Therefore, in this study we introduce a modified modularity matrix **V,** which ranks the ShReD between two reaction nodes relative to the distribution of all ShReDs involving either one of the two reaction nodes. There is a positive correlation ($R^2=0.35$, $p<0.01$) between the un-weighted ShReD and the corresponding weighted ShReD. The correlation analysis was performed on reaction pairs with ShReD < 100, since the maximal ShReD value was capped at 100 (see Methods). The positive correlation suggests that the topology of the metabolic network as defined by the stoichiometry has some influence on the closeness of cyclical interactions between enzymes as defined by the fluxes of the reactions connecting the enzymes (Figure 4-8).

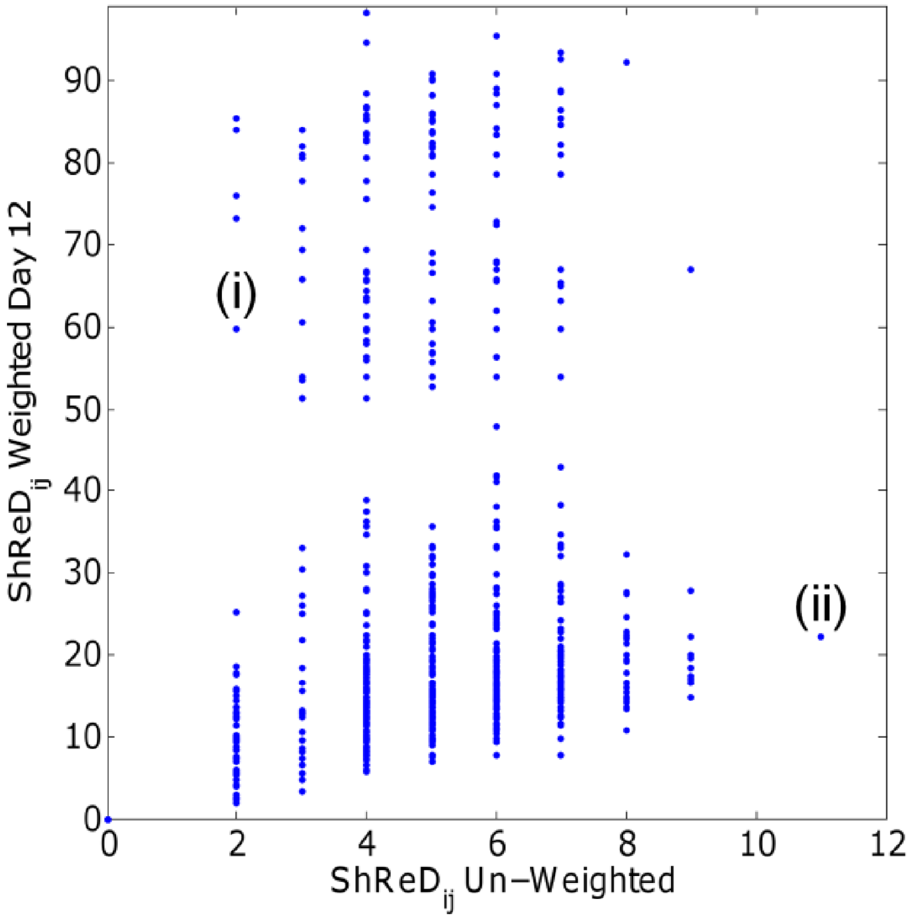


Figure 4-8: Comparison of un-weighted vs. weighted ShReDs.

Scatter plot of Day 12 weighted ShReD and un-weighted ShReD for all reaction pairs [I,j]. The plot only shows reaction pairs with ShReD value < 100. Points (i) and (ii) represent reaction pairs [R28,R64] and [R43,R50].

However, the correlation is not very strong, as there are many instances where a relatively short un-weighted ShReD corresponds to a relatively long weighted ShReD (Figure 4-8, i), and a long un-weighted ShReD corresponds to a short weighted ShReD (Figure 4-8, ii).

4.4.2 Effects of Edge-Weighting on ShReD-based Network Partition

The ShReD-based hierarchical partition of modules for the un-weighted adipocyte model is compared to the partition for the flux-weighted Day 12 model (Figure 4-9). There are striking similarities between the two partitions. In both cases, the transport reactions and a few amino acid metabolism reactions peel off from the original network after the first partition. Additionally, several key interactions between reactions are conserved. The partitions point to close interactions between carbohydrate metabolism, lipid metabolism, and citrate-malate cycle for both the un-weighted case (Figure 4-9A: Module #7288) and the flux-weighted case (Figure 4-9B: Module #7299). Similarly, there is tight coupling between triglyceride synthesis and degradation that persists through multiple partition levels for both cases (Figure 4-9A: Module # 7279, Figure 4-9B: module #7280).

However, there are also several qualitative differences between the two partitions. The un-weighted partition broadly distributes TCA cycle and amino acid reactions across various branches in the hierarchical partition tree (Figure 4-9A). However, in the flux-weighted case, the *a priori* assigned textbook associations largely remain intact (Figure 4-9B: Module #7224, Module #7287). Quantitatively, the flux-weighted partition has a greater average homogeneity index between heights 1-7 in the hierarchy (Figure 4-10), where height zero corresponds to terminal nodes. At height zero, the average homogeneity is similarly high for both the weighted and un-weighted cases due to the large number of single reaction modules.

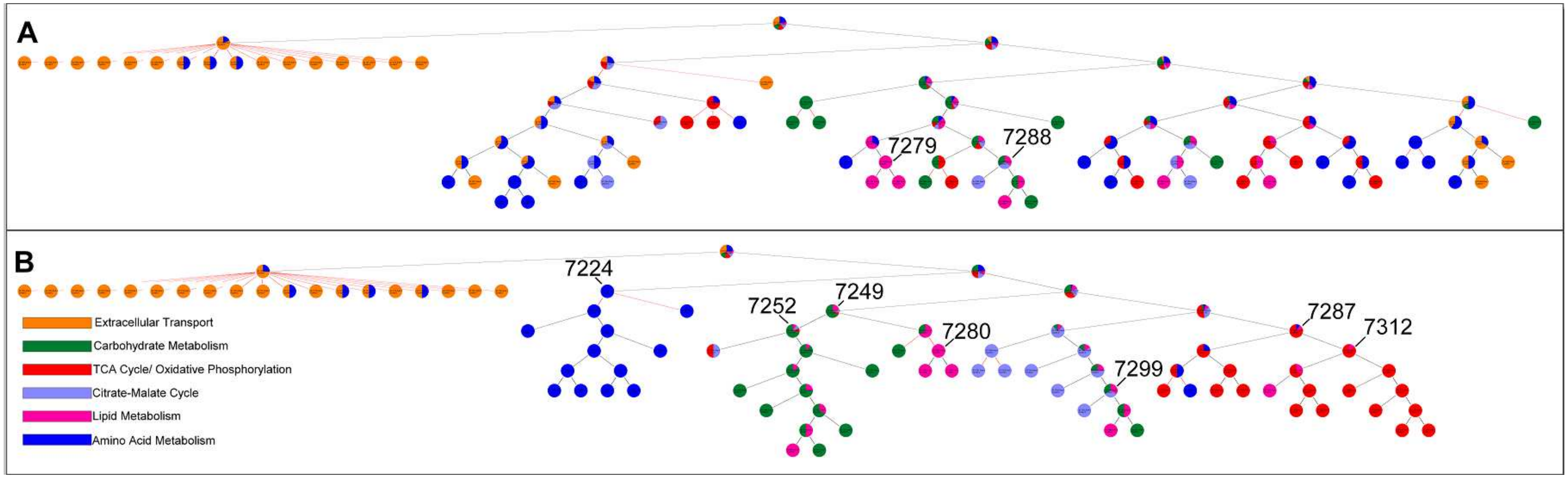


Figure 4-9: Comparison of partitions for unweighted and weighted networks.

4.4.3 Comparing Dynamic vs. Static Weighting Schemes

In the absence of flux data, topological data other than cyclical connectivity could be used to guide modularity analysis. Metabolite degrees (Croes *et al.*, 2006) were investigated as an example of connectivity-based weights reflective of network topology from a static perspective. Briefly, the edge distance from a reaction node R_i to reaction node R_j was determined as the number of reactions in the network that consume the intermediary metabolite connecting R_i and R_j. The rationale was that the influence of R_i on R_j would be strongest if R_j is the only reaction consuming the intermediary metabolite produced by R_i. The influence would be weaker if the intermediary metabolite was consumed not only by R_j, but also by many other reactions in the network. Applying this weighting scheme to the adipocyte model (Figure 4-10), we find that ShReD-based partitioning of the metabolite degree-weighted network results in average homogeneity index values that lie between the un-weighted network and the flux-weighted network.

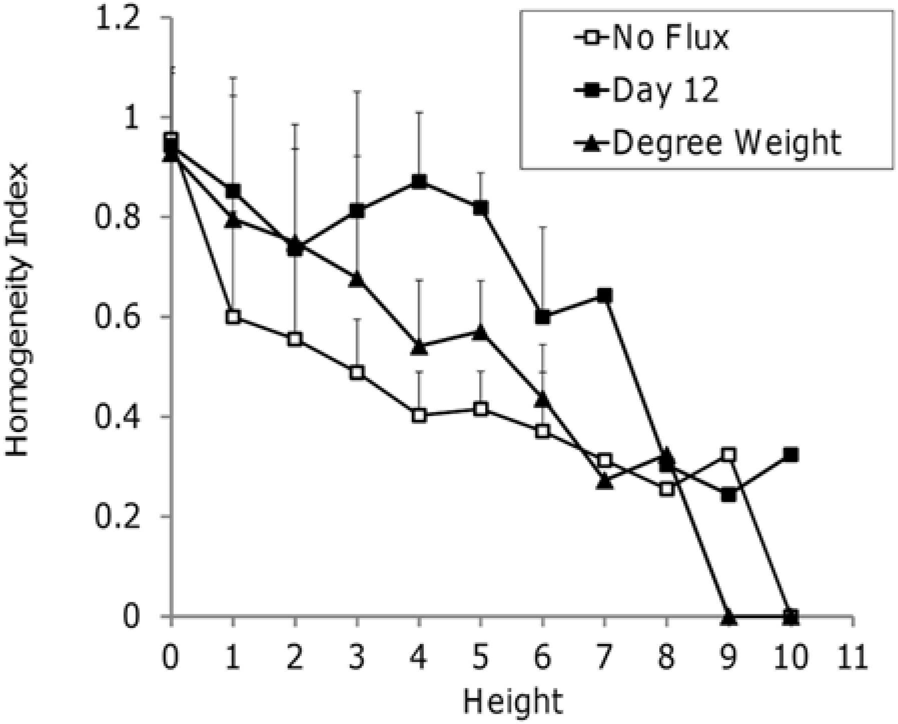


Figure 4-10: Average homogeneity index of modules vs. height

This result suggests that the metabolite degree-weighted network is an improvement over the un-weighted network, but is less effective than the flux-weighted network at capturing the relative engagements between the reactions.

4.4.4 Robust Interaction Pairs

We next investigated whether the modularity score V_{ij} of two reaction nodes in the initial un-partitioned network could predict the degree to which the two reaction nodes remain together in the hierarchical partitioning. The degree to which two reaction nodes remain together was assessed by the partition score H_{ij}, which scales the number of modules shared by both reaction nodes with respect to the total depth of the partitions for each reaction node (see Methods for definition of depth). A scatter plot of the partition score and the modularity score for the Day 12 flux-weighted model shows a significant positive correlation ($R^2 = 0.45$, $p<0.01$) for reaction pairs with a positive modularity score (Figure 4-11).

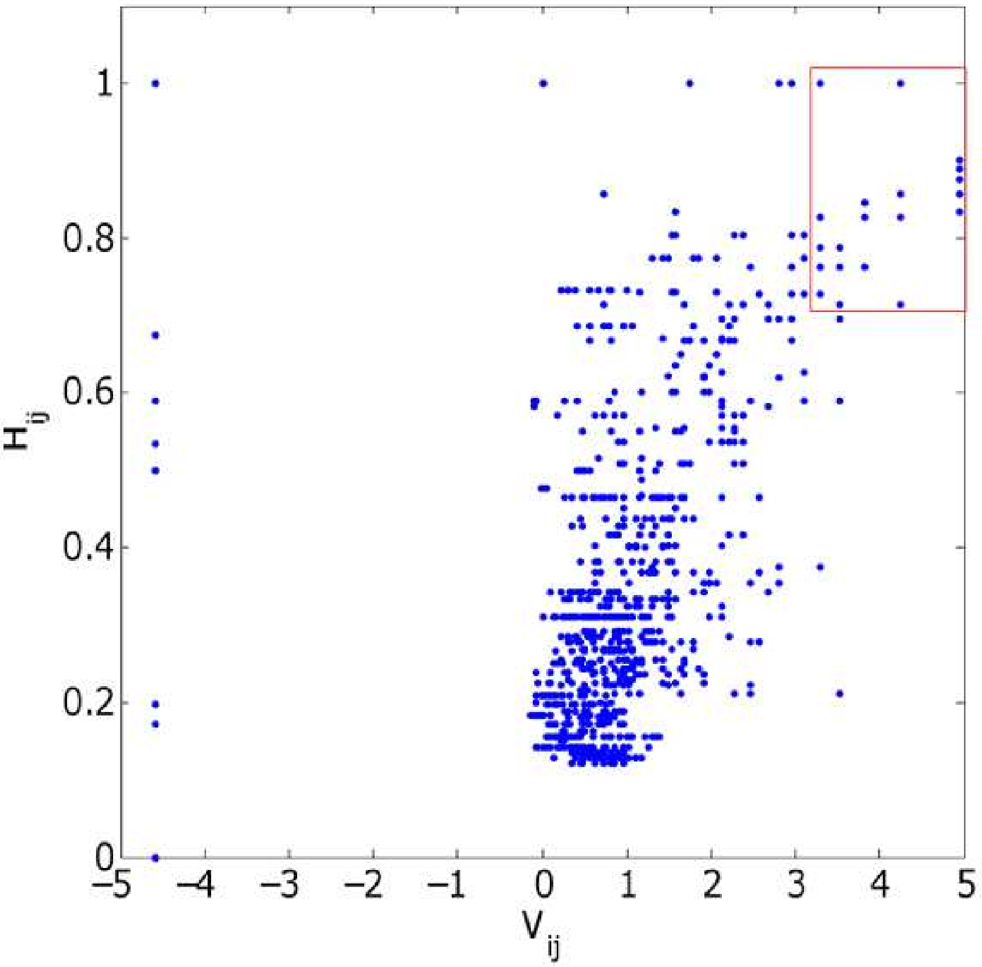


Figure 4-11: Scatter plot of partition score (H) and modularity score (V) for Day 12 model

A red box is drawn around reaction pairs with Hij>0.7 and Vij>3.0

Of particular interest are the reaction pairs that fall in the upper right hand corner, chosen here to be reaction pairs with $V_{ij} > 3.0$ and $H_{ij} > 0.7$. Reaction pairs satisfying this criterion were selected from all four flux-weighted adipocyte models (Day 4, Day 12, and Day 12 with PCX or

LDH inhibition). Forty reaction pairs, or roughly 1.5% of the possible 2556 reaction pairs, satisfied the criterion for at least one of the four models. A heat map displaying the number of models (of the four adipocyte models) for which a given reaction pair meets the criterion shows that 17 of the 40 reaction pairs robustly partition together across the different metabolic states (Figure 4-12).

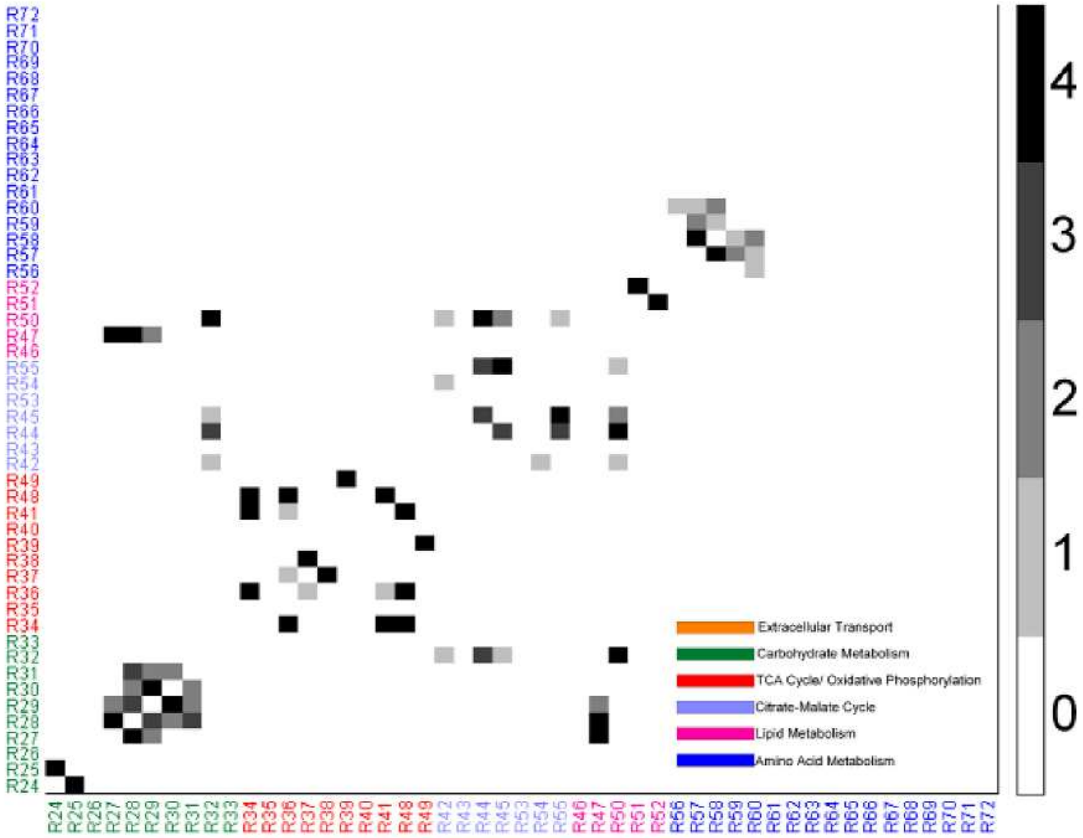


Figure 4-12: Heatmap showing the number of models a reaction pair had Hij>0.7 and Vij>3.0

A reaction pair was designated as robust if Hij>0.7 and Vij>3.0 in all four models. The robust reaction pairs are shown as black squares in the heat map.

One such reaction pair is [R32, R50] (for reaction definitions, see Supplementary Table 1), which corresponds to NADPH production from the pentose phosphate shunt and NADPH consumption for palmitate synthesis, respectively. To determine whether these robust reaction pairs could be identified solely based on stoichiometry in the absence of flux information, each of the 17 reaction pairs were mapped onto a corresponding plot of modularity and partition scores for the un-weighted adipocyte model (Figure 4-13).

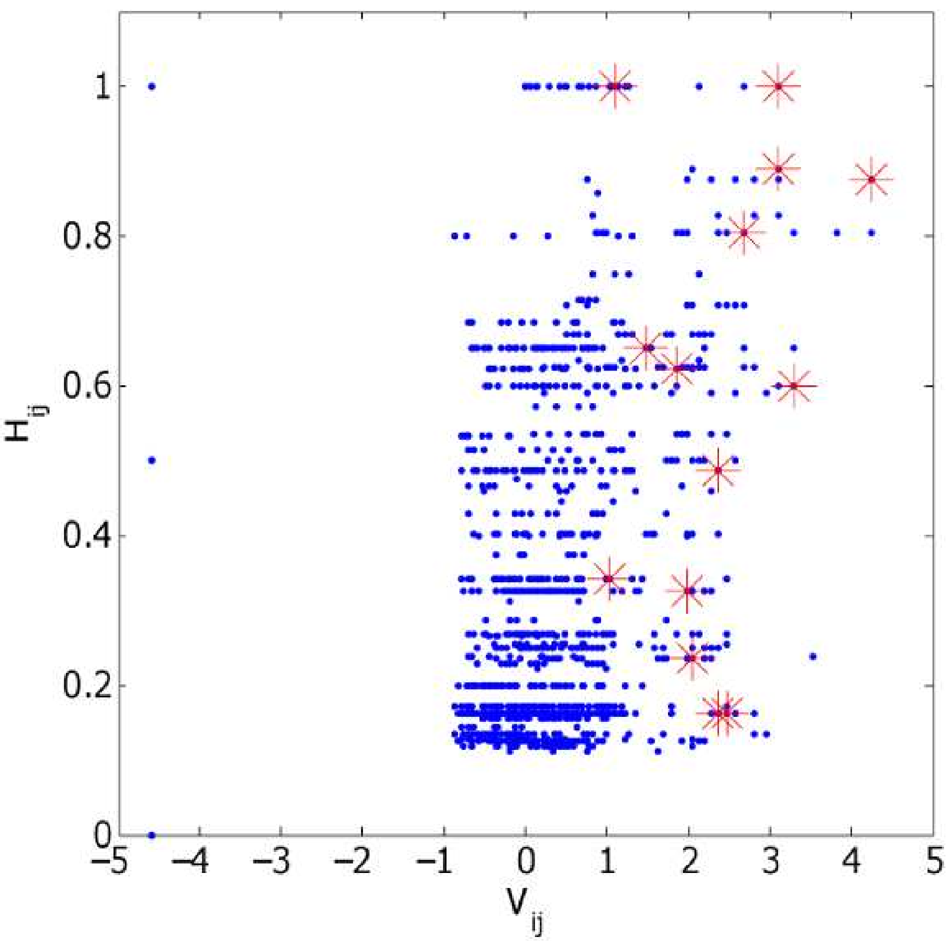


Figure 4-13: Scatter plot of Hij and Vij for all pairs in the un-weighted model.

The red asterisks denote 17 robust reaction pairs identified in Figure 4-12

Overall, the correlation between the partition and modularity scores, albeit still significant, was weaker for the model without flux weights (R^2=0.11, p<0.01). Only 5 of the 17 robust reaction pairs identified in Figure 4-12 have partitions scores > 0.7, and only 3 reaction pairs also have modularity scores > 3.0 in the un-weighted H-V plot. The three reaction pairs are [R34, R36] corresponding to pyruvate dehydrogenase/citrate synthase and isocitrate dehydrogenase in the TCA cycle, [R51, R52], corresponding to triglyceride synthesis and degradation and [R57, R58], corresponding to glutamate synthesis and degradation. The remaining robust reaction pairs identified in the four flux-based partitions are not found in the un-weighted network partition. For example, the robust pair [R29, R30], which corresponds to reactions in glycolysis, has relatively low partition and modularity scores in the un-weighted case.

4.4.5 Impact of Metabolic State on Modularity

The partitions of the four flux-weighted models show clear differences in hierarchical modularity. The differences are mainly due to the placement of the reactions catalyzed by phosphoenolpyruvate carboxykinase (PEPCK) and pyruvate carboxylase (PCX). For the Day 4 model, the first partition of the initial network isolates both enzymes from the main network along with transport reactions that do not have any cyclical interactions (Figure 4-14A: Modules #7337, 7336 respectively).

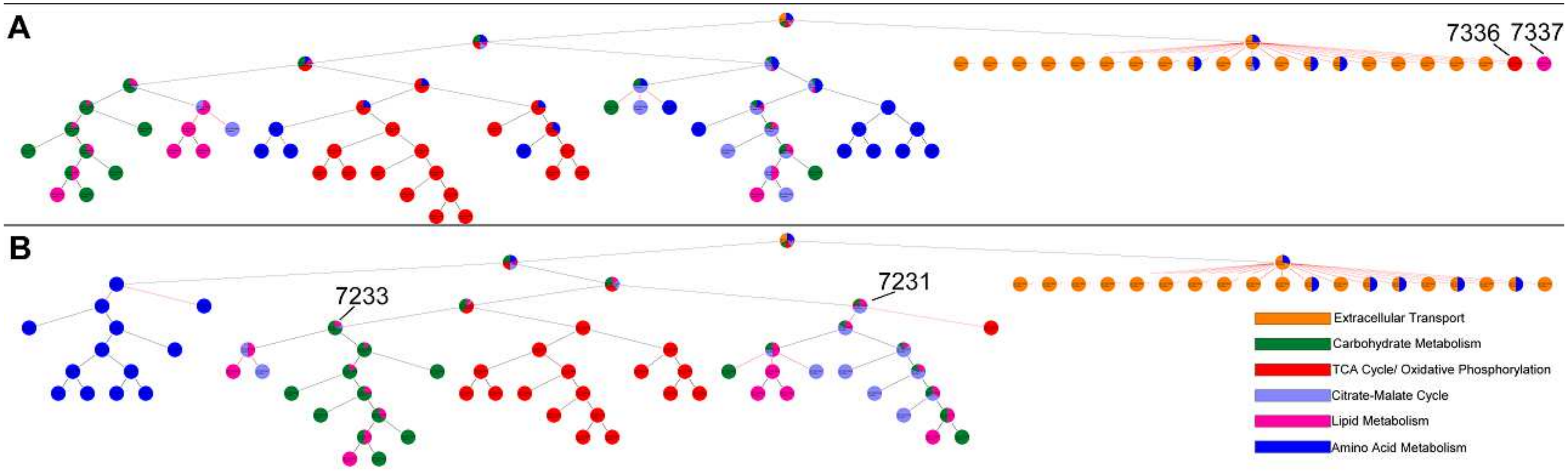


Figure 4-14: ShReD-based partition for (A) Day 4 model and (b) LDH-inhibition model

For the Day 12 model, PEPCK is tightly coupled to the TCA cycle reactions (Figure 4-9B, Module #7312), and PCX is coupled to carbohydrate metabolism (Figure 4-9B, Module #7252). For the LDH-inhibition model, PEPCK is coupled to carbohydrate metabolism reactions (Figure 4-14B, Module #7233) while PCX is coupled to triglyceride metabolism and reactions that produce NADPH (Figure 4-14B, Module #7231).

To quantitatively assess and visualize the overall impact of metabolic state on the hierarchical partitioning of modules, we computed the Euclidean distance a reaction pair moves in the normalized H-V space of a flux-weighted model relative to another flux-weighted model (See Methods). A large distance implies that the reaction pair's partition and modularity scores (the coordinates of the H-V space) are substantially influenced by the change in metabolic state, whereas a short distance suggests that the change in metabolic state has little impact on the reaction pair's placement in the module hierarchy. These distances are shown as heat maps for the changes in flux distribution between Day 4 and Day 12 (Figure 4-15A) and Day 12 and Day 12 with LDH inhibition (Figure 4-15B).

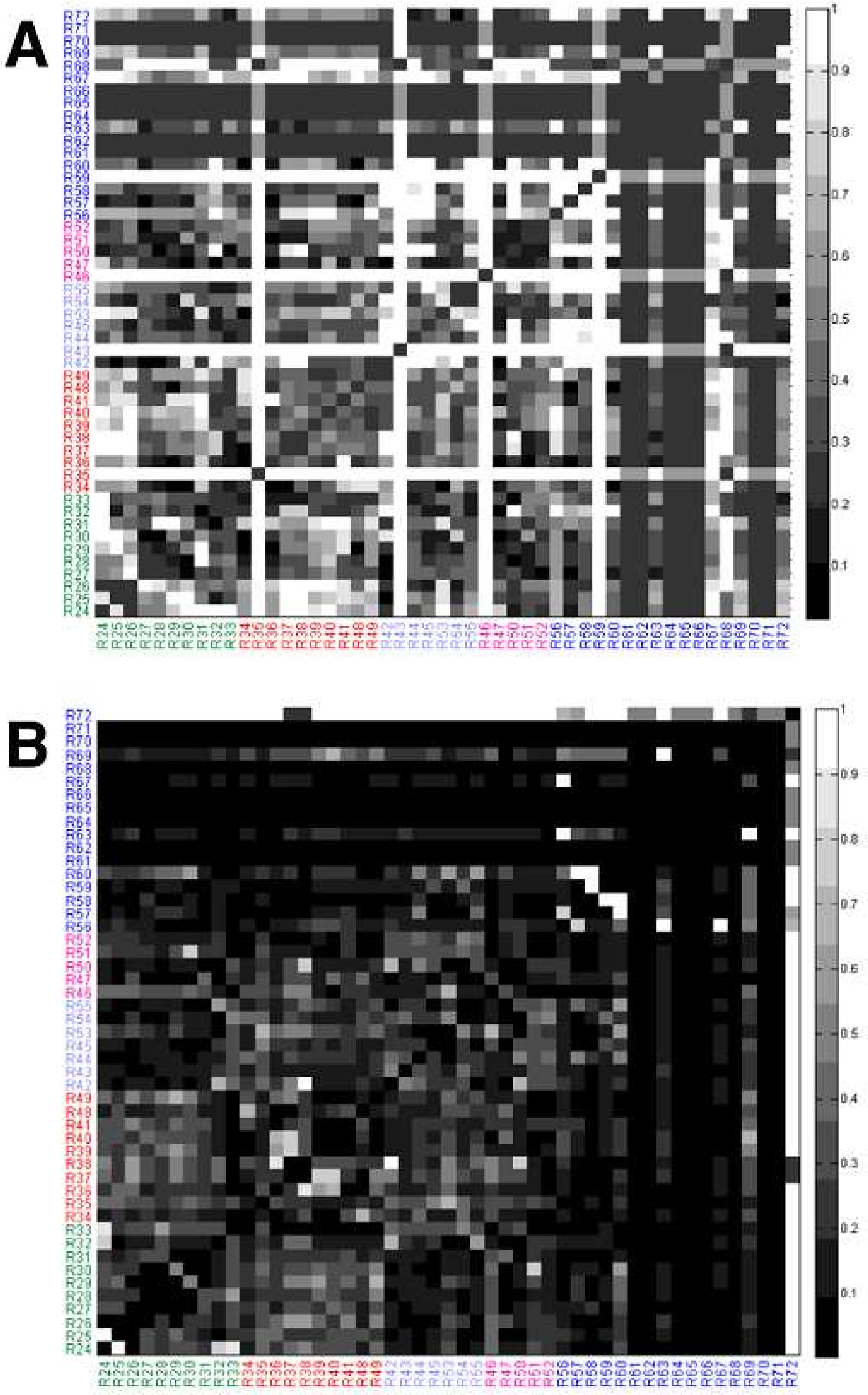


Figure 4-15: Heat map of H-V euclidean distance between two states

(A) Comparison of Day 4 to Day 12 and (B) Day 12 to Day 12 with LDH inhibition

For both models, the heat maps feature several dark regions (reflecting very little movement). These regions largely correspond to reaction pairs involving amino acid metabolism, but also include reaction pairs involved in carbohydrate metabolism (e.g. [R27, R30]), and the TCA cycle (e.g. [R39, R41]). Overall, it is clear that the change in flux distribution between Day 4 and Day 12 has a more pronounced effect on modularity compared to the change in flux distribution between Day 12 and Day 12 with LDH inhibition. A similar observation was made for the change in flux distribution between Day 12 and Day 12 with PCX inhibition, which also had a smaller impact on modularity compared to the change in flux distribution between Day 4 and Day 12 (Fig 4-16).

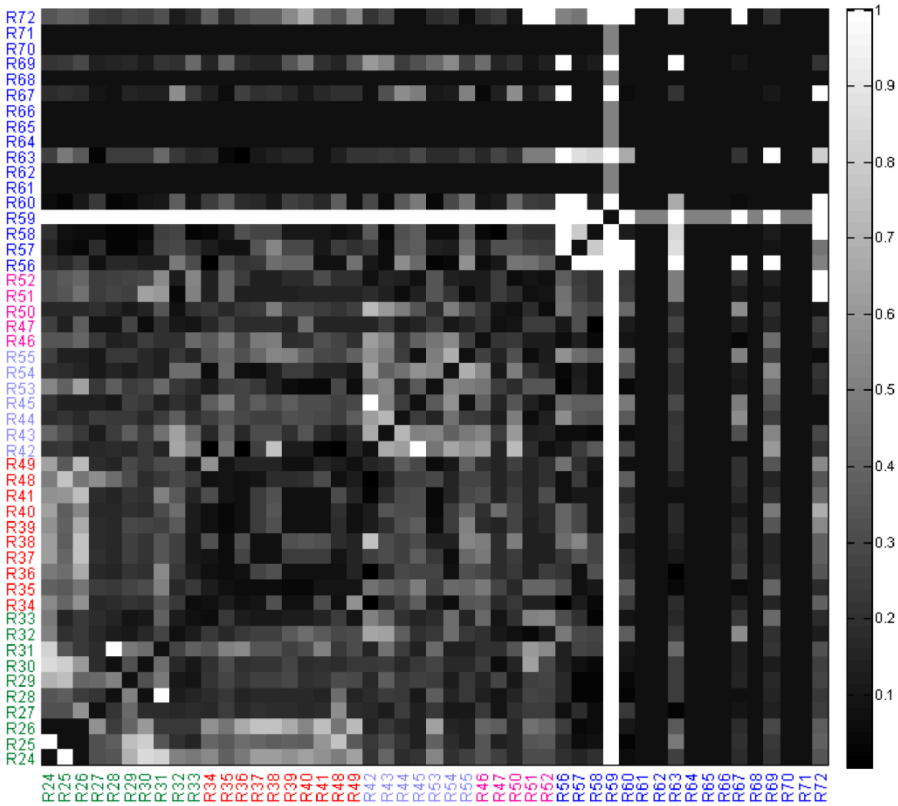
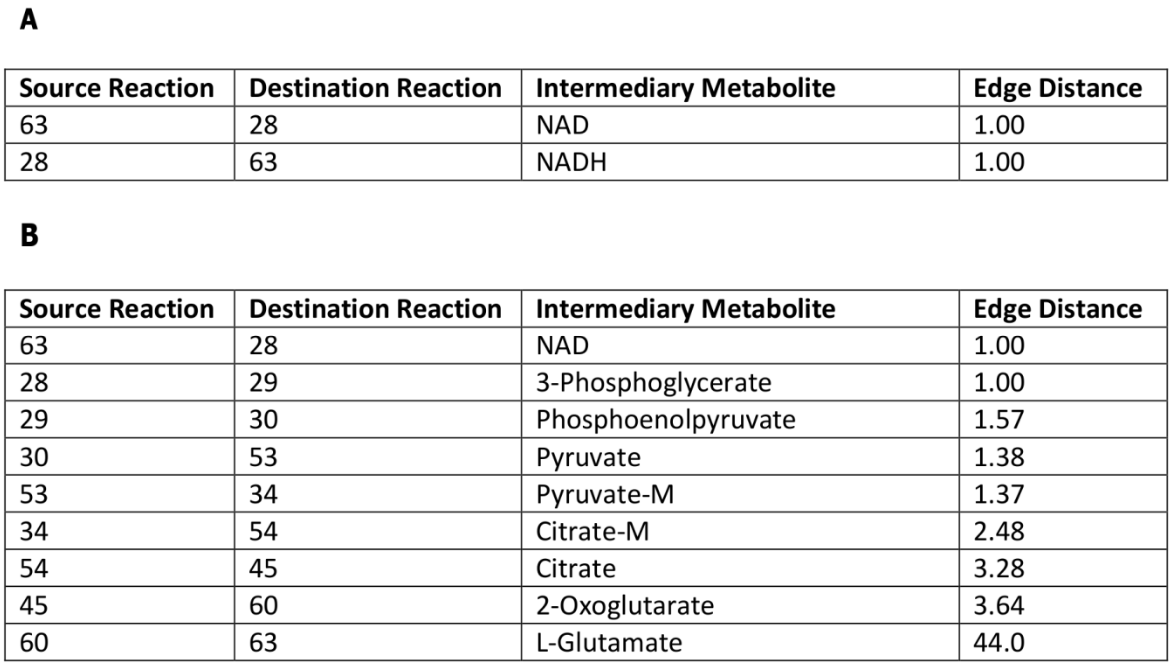


Figure 4-16: Heatmap for H-V Euclidean distance: Day 12 to Day 12 PCX inhibition.

4.5 Discussion

In this chapter, we present a novel methodology for investigating the impact of different metabolic states on the functional organization of metabolic networks. The methodology utilizes metabolic flux data as weights for a graph-based partitioning method that conserves cyclical interactions. Previously, we assessed the cyclical interactions based on ShReDs calculated by assuming static interactions, and thus a uniform graph distance, between each connected reaction pair. In the present study, we allow the interactions, and thus the graph distances, to vary with the metabolic state.

Unlike the un-weighted case, the weighted ShReD distribution displays significant skewness (Figure 4-7), indicating that the arithmetic mean is not representative of the average or expected ShReD. A likely reason for the skewness is that some reactions, particularly those involved in amino acid metabolism, carry negligible flux compared to other parts of central carbon metabolism such as glycolysis and the TCA cycle. As the edge weights of a ShReD are inversely proportional to the fluxes of the reactions comprising the ShReD, a ShReD that includes one or more reactions carrying negligible flux can be very large, and thus skew the arithmetic mean of the distribution. For this reason, it is possible for a reaction pair to have a relatively small ShReD in an un-weighted network, but a relatively large ShReD in a corresponding weighted network. For example, the ShReD for the reaction pair [R28, R63] in the unweighted network is two (2) (Figure 4-8, i), since R28 (3-phosphoglycerate synthesis in glycolysis) and R63 (proline synthesis) interact cyclically via the production and consumption of NADH and NAD^+. However, in the weighted network, the directional interaction from R28 to R63 is very weak, since only a very small fraction of the NADH produced by glycolysis is used for proline synthesis. The corresponding edge distance is 1251, which is approximately 50-times the average of non-infinite ShReDs in the network (25). As a result, the weighted ShReD between these two reactions traverses an alternate sequence of reaction nodes, comprising 10 reactions spanning parts of glycolysis and the TCA cycle, 2-oxoglutarate synthesis, and glutamate synthesis (Figure 4-17).



A

Source Reaction	Destination Reaction	Intermediary Metabolite	Edge Distance
63	28	NAD	1.00
28	63	NADH	1.00

B

Source Reaction	Destination Reaction	Intermediary Metabolite	Edge Distance
63	28	NAD	1.00
28	29	3-Phosphoglycerate	1.00
29	30	Phosphoenolpyruvate	1.57
30	53	Pyruvate	1.38
53	34	Pyruvate-M	1.37
34	54	Citrate-M	2.48
54	45	Citrate	3.28
45	60	2-Oxoglutarate	3.64
60	63	L-Glutamate	44.0

Figure 4-17:Comparison of unweighted (A) and weighted (B) ShReD reaction path.

The ShReD value of this cycle is ~60. This ShReD value is still relatively large compared to other weighted ShReD values in the distribution, implying a relatively weak cyclical interaction. Conversely, a relatively long ShReD in the unweighted network can yield a relatively short ShReD in the weighted network. For example, the unweighted ShReD for the reaction pair [R40, R52], corresponding to mitochondrial malate synthesis and triglyceride degradation respectively, is the largest non-infinite ShReD at 11 (Figure 4-8 ii). However, every edge in this cycle carries a relatively large flux, resulting in a weighted ShReD value of 22, which is close to the average ShReD of the weighted network.

A comparison of the hierarchical partition trees for the unweighted and weighted (Day 12) models shows that the weighted model yields greater functional homogeneity of modules based on the canonical pathway assignments of the constituent reactions (Figures 4-9, 4-10). This suggests that the network topology alone, as defined by the network's reaction stoichiometry, is insufficient to capture the functional associations between reactions that are reflected in the textbook pathway assignments. In the previous chapter, we augmented the stoichiometric information by including known regulatory interactions between reactions. Edges denoting regulatory interactions were drawn from one reaction node to another if the product metabolite of the first reaction allosterically regulated the second reaction. The presence of these regulatory edges had a significant impact on the modularity of the network. However, for many cell types, information regarding regulatory mechanism is incomplete or difficult to obtain, requiring extensive manual searches of the literature. Therefore, an un-weighted network will almost certainly contain only partial information regarding functional interactions between reactions. One way to upgrade the information content is to incorporate metabolic flux data, which provides a snapshot of cellular metabolic state, and reflects the integral of various regulatory processes active in the cell. In this study, we found that incorporating flux data as weights for directed interactions between reactions resulted in homogeneous modules that are more in line with textbook knowledge on biochemical pathway organization.

However, we found that some module inhomogeneity persists deep into the hierarchy even for the weighted models. A majority of these inhomogeneous modules include one or more robust reaction pairs that consistently partition together across the different metabolic states examined in this study. One such module, found at depth 7 of the Day 12 model partition (Figure 4-9B, Module #7299), points to a tight coupling between carbohydrate metabolism, citrate malate cycle, and lipid metabolism, mediated through the production and consumption of NADPH. This module includes the reaction pair [R32, R50], corresponding to NADPH production via the pentose phosphate shunt and palmitate synthesis respectively, which was one of the 17 robust reaction pairs with both a high modularity score and a high partition score for all four flux-weighted partitions. We have previously observed that the interactions mediated by cofactors, which are ubiquitously present throughout metabolism, can couple reactions spanning seemingly distant pathways. Prior studies have often removed cofactors or 'currency metabolites' prior to network modularization due to the difficulty of assigning them to distinct functional modules. While ShReD-based partitioning can also be performed after the removal of cofactors, our prior work suggests that cofactors are essential in mediating metabolic cycles and allosteric feedback loops, and should thus be retained if the goal is to identify modules based on cyclical interactions.

One possible biochemical basis underlying the robust reaction pairs is co-regulation. For example, reaction R50, catalyzed by 3-oxoacyl-(acyl-carrier-protein) reductase, requires NADPH as a cofactor for activity (Carlisle-Moore *et al.*, 2005), while both enzymes catalyzing the lumped reaction R32, glucose 6-phosphate dehydrogenase and 6-phosphogluconate dehydrogenase, are allosterically regulated by NADPH (Ozer *et al.*, 2001; Rippa *et al.*, 1998). Similarly, reaction R44, catalyzed by malic enzyme, is product-inhibited by NADPH (Shearer *et al.*, 2004), which is a required cofactor for reaction R50. Another co-regulated robust reaction pair is [R34, R48], corresponding to the first steps in the TCA cycle (pyruvate dehydrogenase and citrate synthase) and oxidative phosphorylation, respectively. Oxaloacetate is a limiting substrate for citrate synthase, and also a competitive inhibitor of oxidative phosphorylation (Dervartanian and Veeger, 1964). Reactions R34 (pyruvate dehydrogenase/citrate synthase), R36 (isocitrate dehydrogenase) and R41 (malate dehydrogenase) are steps in the TCA cycle regulated by ATP, which could explain the robustness of interactions between reaction pairs [R34, R36] and [R34, R41] (Shearer *et al.*, 2004; Nelson and Cox, 2008; Martinez-Rivas and Vega, 1998). While the partitions of the four flux-weighted models share similar modules as exemplified by the robust reaction pairs, they also exhibit notable differences. For the Day 4 partition, reactions catalyzed by PCX and PEPCK both split off immediately from the parent network at depth one of the hierarchy. This split is due to the very low flux carried by these reactions at Day 4, which excludes them from significant cyclical interactions with any of the other reaction nodes. Day 4 represents an early stage of differentiation when an immature adipocyte phenotype is expected. While lipogenic genes are activated, the fluxes of lipid synthesis and triglyceride accumulation remain low at this stage relative to other parts of central carbon metabolism. Our results suggest that PCX and PEPCK, which catalyze upstream steps in glycerogenesis and fatty acid synthesis from glucose, are not yet integral to any major functional modules in the immature adipocyte. However, at Day 12 (Figure 4-9B), PEPCK is tightly coupled to the TCA cycle reactions, mediated through the consumption and production of ATP, and PCX is coupled to carbohydrate metabolism and triglyceride metabolism (Figure 4-9B Module #7252). Indeed, there is a striking difference between Day 4 and Day 12 partitions based on the relative distances between the corresponding pairs of reactions in the H-V space. In comparison, there is a more subtle difference between the partitions of Day 12 and Day 12 with LDH inhibition. These observations suggest that the inhibition of one enzyme is not enough to drastically alter modularity. In contrast, the transition from an immature phenotype on Day 4 to a mature phenotype on Day 12 represents a concerted set of changes across cellular metabolism, which is reflected in the broadly altered modularity.

4.6 Conclusions

Taken together, our results support the notion that network modularity is influenced by both the connectivity of the network's components as well as the relative engagements of the connections. The major contribution of this chapter is a generally applicable methodology to incorporate activity data into a systematic partitioning framework featuring the conservation of cyclical, or retroactive, interactions. We found two key benefits of incorporating metabolic flux data. First, comparisons across different metabolic states can identify conserved modules comprising robustly interacting reactions that may be co-regulated by a common allosteric effector. Second, embedded in the flux data is information on the various layers regulatory processes active in the cell, which can be used to augment connectivity relationships defined by stoichiometry. In the context of modularity analysis, the implication is that lack of detailed knowledge on regulatory mechanisms can be at least partially addressed using experimentally observable data. On the other hand, the reliance on experimental data is also a limitation in the scalability of our methodology. As modularity analysis is an approach to study complex networks, it is ideally applied to large-scale systems rich with complexity. Unfortunately, resolving the flux distribution of a large-scale metabolic network, for example using ^{13}C isotope labeling, remains experimentally demanding and technically challenging. One way to address this limitation in scalability could be to utilize solutions from constraint-based methods such as Flux Balance Analysis that require relatively few measurements. Rather than rely on flux data reflecting an observed metabolic state, flux data could be used that reflect an optimized state or a range of attainable states. An added benefit of using such model-derived flux data could be to enable efficient exploration of different module configurations accessible to a metabolic network

In summary, we have extended our previously developed methodology for ShReD-based modularity analysis by considering non-uniform interactions between retroactively connected reactions. Whether the modules defined by cyclical interactions between their constituent reactions indeed contribute to some recognizable system property warrants further study. In the next chapter, we investigate the presence of substrate cycles within ShReD-based modules, as we have shown that such a motif may serve to limit the propagation of perturbations through the network, and thereby add to the stability of the system.

5 Modularity Analysis Guides Experimentation Using Targeted Metabolomics

5.1 Abstract

The modularity of a metabolic network based on Shortest Retroactive Distances revealed tightly coupled interactions between reactions that may not be necessarily apparent based on a two dimensional cartography of metabolism. Moreover, it has been shown that the modularity of the system is state-dependent when the relative engagements between the reactions are taken into account. This theoretical framework for mapping the functional organization of the cellular system can guide hypotheses regarding how metabolism would be affected if a particular module or several modules were perturbed. These hypotheses can then be tested experimentally by measuring the appropriate changes in the concentration of metabolites, consumed or produced by reactions involved in the perturbed module. As such, *in silico* predictions leads to a targeted analysis of specific metabolite concentrations, thus invoking the experimental work flow of targeted metabolomics using tandem mass spectrometry (LC/MS-MS). In this chapter, we highlight an experimental biological application where targeted metabolomics is used to monitor changes in metabolism of the gut microbiota as a function of mice age. We focus on tryptophan metabolism, and investigated the *in vivo* concentrations of gut microbiota derived metabolites with tryptophan as the substrate. At the completion of this work, metagenomic networks incorporating microbiota sequencing data were not available to partition and identify ShReD-based modules containing tryptophan metabolism reactions. We therefore used a probabilistic search tool to identify microbiota produced tryptophan-derived metabolites

5.2 Introduction

The human gastrointestinal (GI) tract is colonized by ~10^{14} bacteria belonging to ~1,000 species that are collectively termed the microbiota. Disruptions in the microbiota composition (dysbiosis) are increasingly correlated to not only gut diseases such as inflammatory bowel disease (IBD) and colitis (Chassaing and Darfeuille-Michaud, 2011), but also obesity, insulin resistance and type 2 diabetes (Burcelin *et al.*, 2011; Turnbaugh and J. I. Gordon, 2009). There is increasing evidence that the functional outputs of the microbiota, specifically the metabolites they produce, are important modulators of host physiology. For example, recent work from our laboratory demonstrated that the tryptophan (TRP)-derived bacterial metabolite indole attenuates indicators of inflammation and improves tight junction properties in intestinal epithelial cells (Bansal *et al.*, 2010). An increase in the conversion of dietary choline into trimethylamine by the gut microbiota has also been correlated to non-alcoholic fatty liver disease (Dumas *et al.*, 2006) and cardiovascular diseases (Z. Wang *et al.*, 2011).

Despite a high level of interest, only a handful of bioactive microbiota metabolites in the GI tract have been identified. One major challenge is that the spectrum of metabolites present in the GI tract is extremely complex, as the microbiota can carry out a diverse range of biotransformation reactions, including those that are not present in the mammalian host (Van Duynhoven *et al.*, 2011). Classical approaches such as isolating and culturing individual bacteria and identifying metabolites produced in these cultures has not yielded much success, as many bacterial species in the GI tract cannot be cultured under standard laboratory conditions. Moreover, this approach also does not account for community-level interactions between the bacteria as metabolites produced by one bacterial species can be utilized or modified by other species resident in the local microenvironment. Another challenge is that it is also extremely difficult to determine whether a metabolite is the product of microbiota or host metabolism, as most metabolite classes are present in both bacteria and mammals due to the high degree of conservation of metabolic pathways across organisms.

In addressing these challenges, metabolomics of fecal or bodily fluid samples has emerged as an attractive approach to explore the metabolite profiles of the GI tract, and to compare these profiles under different conditions. Mass spectrometry (MS)-based approaches have been especially useful in high-throughput identification of a broad spectrum of metabolites. For example, a recent study by Zheng and co-workers (Zheng *et al.*, 2011) used chromatographic separation coupled with MS to characterize the impact of antibiotic treatment on the metabolome of rat fecal and urine samples, and observed that the levels of more than 200 metabolites were significantly altered. Interestingly, TRP-derived compounds such as indole and tryptamine were among the significantly altered metabolites in both fecal and urine samples. In a related study, Antunes and co-workers (Antunes *et al.*, 2011) used Fourier transform ion cyclotron resonance MS to detect more than 2,000 metabolite features in murine fecal samples, and found that a single high dose of streptomycin causes significant changes in 88% of these features. In addition to MS, high-resolution NMR spectroscopy has also been used to broadly profile the metabolites whose levels in feces, bodily fluids or host tissues are significantly altered by interventions (such as bariatric surgery) that are expected to perturb the gut microbiota (J. V Li *et al.*, 2011). In general, MS offers greater sensitivity compared to NMR, and has become the dominant analytical platform for metabolomics (Lu *et al.*, 2009).

To date, a majority of MS studies on profiling GI tract metabolites have utilized an untargeted approach to achieve high throughput. While this approach offers the benefit of potential for discovery, it also has several drawbacks. Due to the complexity of the mass spectra obtained, especially when full scan tandem mass spectrometry (MS/MS) is employed, metabolite identification can be difficult, as it is difficult to discriminate between ions and ion fragments having the same mass-to-charge ratio (m/z). High-resolution time-of-flight (TOF) mass spectrometers can somewhat alleviate this problem (Lu *et al.*, 2009). However, quantification remains a challenge, because the dynamic range of different metabolites in a biological sample can span up to nine orders of magnitude (Want *et al.*, 2005), and an untargeted approach precludes tailoring of MS parameters for ionization and fragmentation of specific, low-abundance metabolites. In contrast, targeted approaches, where the analytes are determined *a priori*, can use quantitative methods such as multiple reaction monitoring (MRM) that afford custom optimization of MS parameters for individual metabolites to enable sensitive detection. On the other hand, a targeted approach is limited in its discovery potential, as only a focused set of metabolites is analyzed simultaneously. Expanding the number of MRM transitions, and thus detecting more analytes in a single run, requires a reduction in the dwell time to preserve peak shape, which can compromise sensitivity (Lu *et al.*, 2009).

In this work, we present a novel targeted metabolomics methodology that addresses the inherent discovery limitation by integrating an *in silico* prediction step into the MS workflow to identify bioactive microbiota metabolites. To date, bioinformatics tools have been utilized in metabolomics generally for *post-hoc* analysis to process raw data (Brown *et al.*, 2011), or perform statistical comparisons (Martin *et al.*, 2010). Recently, Greenblum and co-workers presented an elegant metagenomic study that places obesity or IBD-associated variations in human gut microbiota gene abundances in the context of a microbial community-level metabolic network reconstructed *in silico* (Greenblum *et al.*, 2011). Here, we similarly model microbiota metabolism using a metabolic network representing a single “microbial metabolic organ” to computationally explore the products of microbiota metabolism. We thus exploit efficient computational algorithms for network analysis and the growing catalogue of annotated microbial genomes to conduct *in silico* discovery experiments. As a result, the analytical effort can focus on a relatively small number of metabolites to support facile quantitation. To validate our methodology, we use computational pathway analysis to predict bacterial products of tryptophan (TRP) metabolism, and utilize MRM coupled with liquid chromatography (LC) to quantify the levels of the predicted metabolites in murine cecum and feces. To determine bioactivity of the metabolites, we use a *Gaussia luciferase* (Gluc) reporter system measuring aryl hydrocarbon receptor (AhR) activation. Our results have the potential to facilitate mechanistic studies on host-bacteria interactions in the intestinal tract.

5.3 Materials and Methods

5.3.1 Materials

All chemicals including HPLC-grade solvents and high-purity metabolite standards were purchased from Sigma-Aldrich (St. Louis, MO) unless noted otherwise.

5.3.2 Metabolite Extraction

Metabolites were extracted from cecum or fecal pellets luminal contents and fecal samples using a solvent-based method (Sellick *et al.*, 2010) with minor modifications. Briefly, 1.5 ml of ice-cold methanol/chloroform (2:1, v/v) was added to a sample tube containing a pre-weighed luminal content or fecal sample. After homogenization on ice, the sample tube was centrifuged under refrigeration (4 ^{o}C) at 15,000 × *g* for 10 min. The supernatant was then transferred to a new sample tube through a (70-μm) cell strainer. After adding 0.6 mL of ice-cold water, the sample tube was vortexed vigorously and centrifuged under refrigeration (4 ^{o}C) at 15,000 × *g* for 5 min to obtain phase separation. The upper and lower phases were separately collected into fresh sample tubes with a syringe, taking care not to disturb the interface. To improve signal intensity for MS, 500 μL of the polar phase was concentrated by solvent evaporation in a Savant speedvac concentrator (Thermo Scientific, Asheville, NC), and then reconstituted in 50 μL of methanol/water (1:1, v/v). Extracted metabolites were stored at -80^{o}C until analysis.

5.3.3 Metabolite Analysis using LC/MS-MS

Prior to sample analysis, MS parameters were optimized for each target metabolite to identify the MRM transition (precursor/product fragment ion pair) with the highest intensity under direction injection at 10 μL/min. The following parameters were optimized operating in positive mode: declustering potential (DP), entrance potential (EP), collision energy (CE), and collision cell exit potential (CXP). The optimized parameter values for the target metabolites analyzed in this study are shown in *SI Appendix* Table S1.

Table 3: Optimized MS parameters for indole and indole-derivatives.

Compound	**Precursor (Da)**	**Product (Da)**	**DP (V)**	**EP (V)**	**CE (V)**	**CXP (V)**
Indole	118.0	91.0	41.0	10.0	27.0	2.5
Indole 3-acetate	176.0	130.0	31.0	9.0	19.0	4.0
Indole 3-acetamide	175.0	130.0	26.0	10.0	19.0	4.0
Tryptamine	161.0	144.2	11.0	4.0	15.0	4.0
Hydroxyindole	134.1	77.1	26.0	10.0	37.0	2.5
Tryptophan	205.0	188.0	21.0	10.0	17.0	4.0

The target metabolites in samples were detected and quantified on a triple quadrupole linear ion trap mass spectrometer (3200 QTRAP, AB SCIEX, Foster City, CA) coupled to a binary pump HPLC (1200 Series, Agilent, Santa Clara, CA). Peak identification and integration were performed using Analyst software (version 5, Agilent, Foster City, CA). Samples were maintained at 4^{o}C on an autosampler prior to injection. Chromatographic separation was achieved on a hydrophilic interaction column (Luna 5 µm NH_2 100 Å 250 mm × 2 mm, Phenomenex, Torrance, CA) using a solvent gradient method (Bajad *et al.*, 2006). Solvent A was an ammonium acetate (20 mM) solution in water with 5 % acetonitrile (v/v). The pH of solvent A was adjusted to 9.5 immediately prior to analysis using ammonium hydroxide. Solvent B was pure acetonitrile. Injection volume was 10 µL. The gradient method, including solvent flow rate and composition, is shown in *SI Appendix*, Table S2.

Table 4: LC Gradient Profile

Time (min)	**Flow rate (µL/min)**	**A (%)**	**B (%)**
0	300	15	85
15	300	100	0
28	300	100	0
30	300	15	85
50	300	15	85

5.3.4 Partition of Metabolites in Extraction Solvent

Since only the upper (polar) phase of the biphasic metabolite extract was used for analysis, the partition of all five metabolites between the two phases was determined for absolute quantification. Briefly, 5 µL of the metabolite's stock solution (1 mg/mL) was dispensed into a 1.5 mL microfuge tube. To this pure stock solution, 750 µL of 2:1 methanol:chloroform was added, followed by 300 µL of water. Following centrifugation and phase separation, the upper phase was collected and analyzed for the target metabolite using the corresponding optimized MRM parameters. The difference between measured and theoretical amounts of the metabolite was used to determine the partition between phases. For example, the theoretical amount of tryptophan in the stock solution-solvent mixture is 25 nmol. The measured concentration in the upper phase was 34 µM, which corresponds to an estimated total amount of ~27 nmol in the upper phase (volume 800 µL), suggesting that all of the tryptophan is recovered within a margin of (e.g. pipetting) error. Similarly, all other metabolites analyzed in this study showed full partition into the upper phase, except indole. For indole, of the 42 nmol dispensed, only 17 nmol was recovered in the upper phase, indicating that the actual amount of indole extracted from the tissue is ~2.5 times greater than the measured amount.

5.3.5 Statistical Analysis

Comparisons of medians between the metabolite levels of young and old mice were performed with the non-parametric two-sided Mann-Whitney *U*-test. The null hypothesis that the two medians are the same was rejected for $p < 0.05$.

5.4 Results

5.4.1 Predicted Metabolite Set

A probabilistic search of possible biochemical transformations was conducted with tryptophan as the source metabolite using a previously published algorithm (Yousofshahi *et al.*, 2011) run in reverse. The search was limited to only reactions not present in murine metabolism, thus producing a list of high frequency metabolites that can potentially be produced by the gut microbiome using tryptophan as the source (Unpublished Result, Lee Laboratory, Figure 5-1).

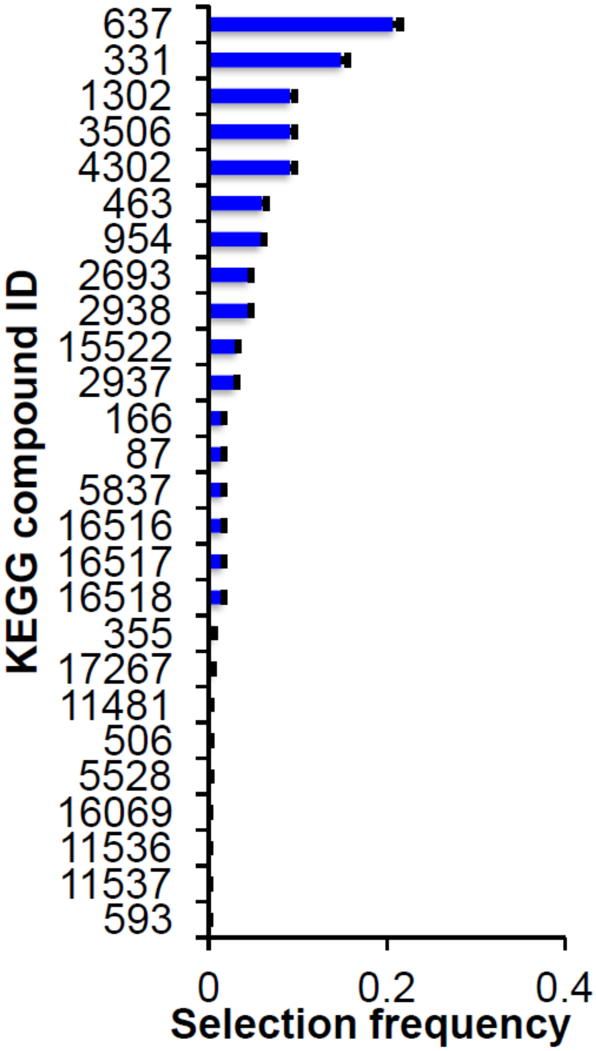


Figure 5-1: High frequency metabolites: *in silico* prediction of tryptophan derived metabolites

5.4.2 Quantification of Metabolites using Multiple Reaction Monitoring (MRM)

Of the 12 high-frequency metabolites predicted by the pathway algorithm to be derived from TRP by bacteria, a subset was selected for MRM analysis based on availability of pure standards and ease of ionization and fragmentation. For example, indole 3-acetaldehyde was excluded, because a high-purity was not available to generate a standard curve. Indole 3-pyruvate was excluded, because it ionized poorly, and thus had a high limit of detection (LOD) in comparison with the other metabolites targeted for analysis. The final panel of metabolites targeted for analysis comprised indole, hydroxyindole, indole 3-acetate, indole 3-acetamide, tryptamine and TRP. Hydroxyindole was added to the panel based on recent reports suggesting that it is derived from indole through bacterial reactions (21). All six metabolites were simultaneously identified and quantified in murine fecal and cecum extracts using MRM MS following separation by hydrophilic interaction chromatography (HILIC) (*Materials and Methods*). Metabolite identification was performed based on both chromatographic retention time and mass signatures, as we found that even an optimized MRM transition (precursor-product ion pair) did not always uniquely identify a metabolite in a complex biological sample. For indole, the optimal ion pair transition based on MS signal intensity was 118/91 m/z. The elution time of this transition ranged from 2.2 and 2.9 min for standards containing the corresponding high-purity chemical (Figure 5-2).

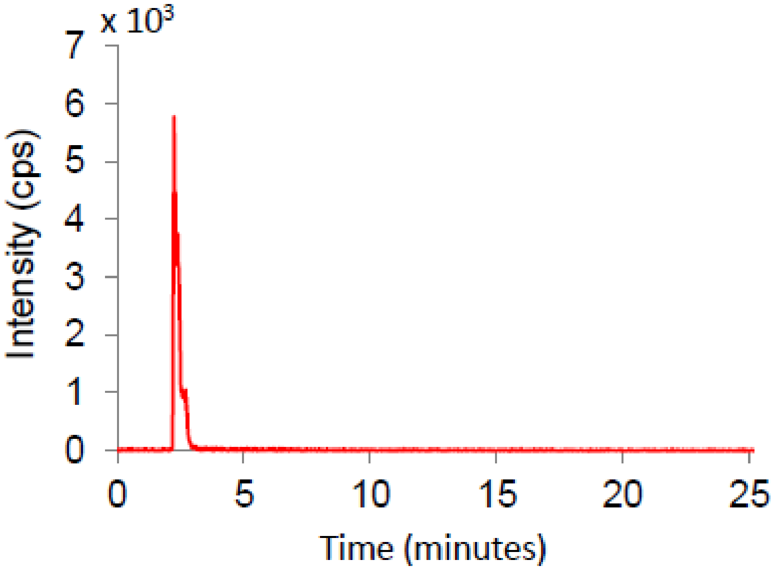


Figure 5-2: Indole Peak for MRM 118>91 for 10^7 ng/L standard.

The linear response of signal area under curve (AUC) with respect to concentration is shown for concentrations up to 85 µM (Figure 5-3; $R^2 = 0.998$, $p < 10^{-4}$), with an estimated LOD (defined as the peak height of 10-times the baseline signal) of 4.6 µM.

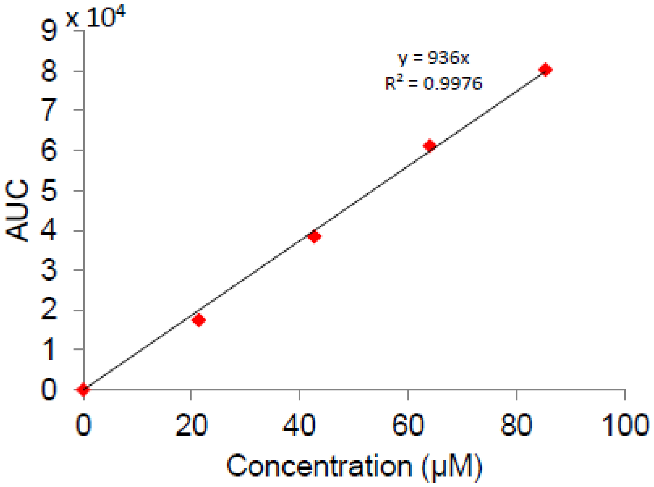


Figure 5-3: Linear response of Indole peak MRM AUC vs Concentration

For tissue extracts, we observed multiple peaks corresponding to the signal-optimized indole transition, highlighting the importance of matching the retention time to unambiguously identify a metabolite (Figure 5-4, Figure 5-5 for chromatograms of all metabolites).

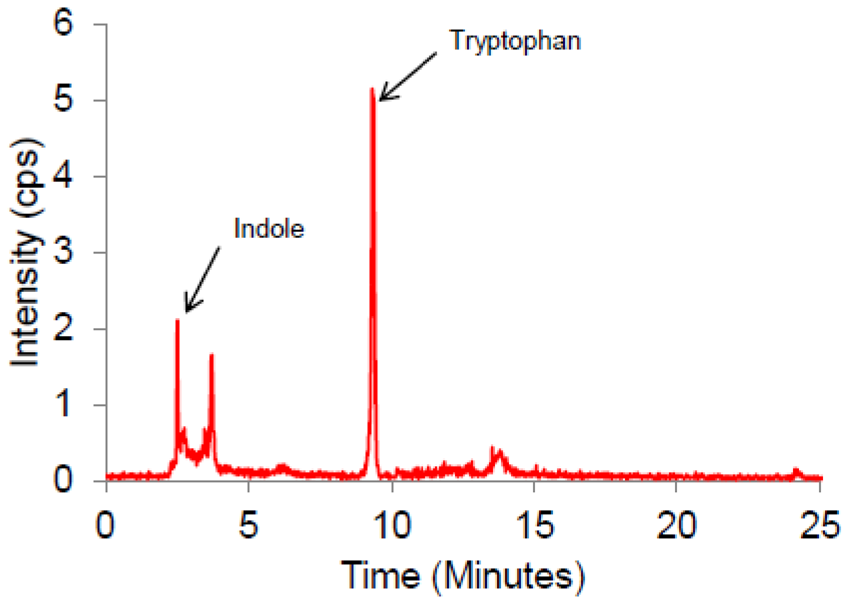


Figure 5-4: Indole MRM in cecum extract

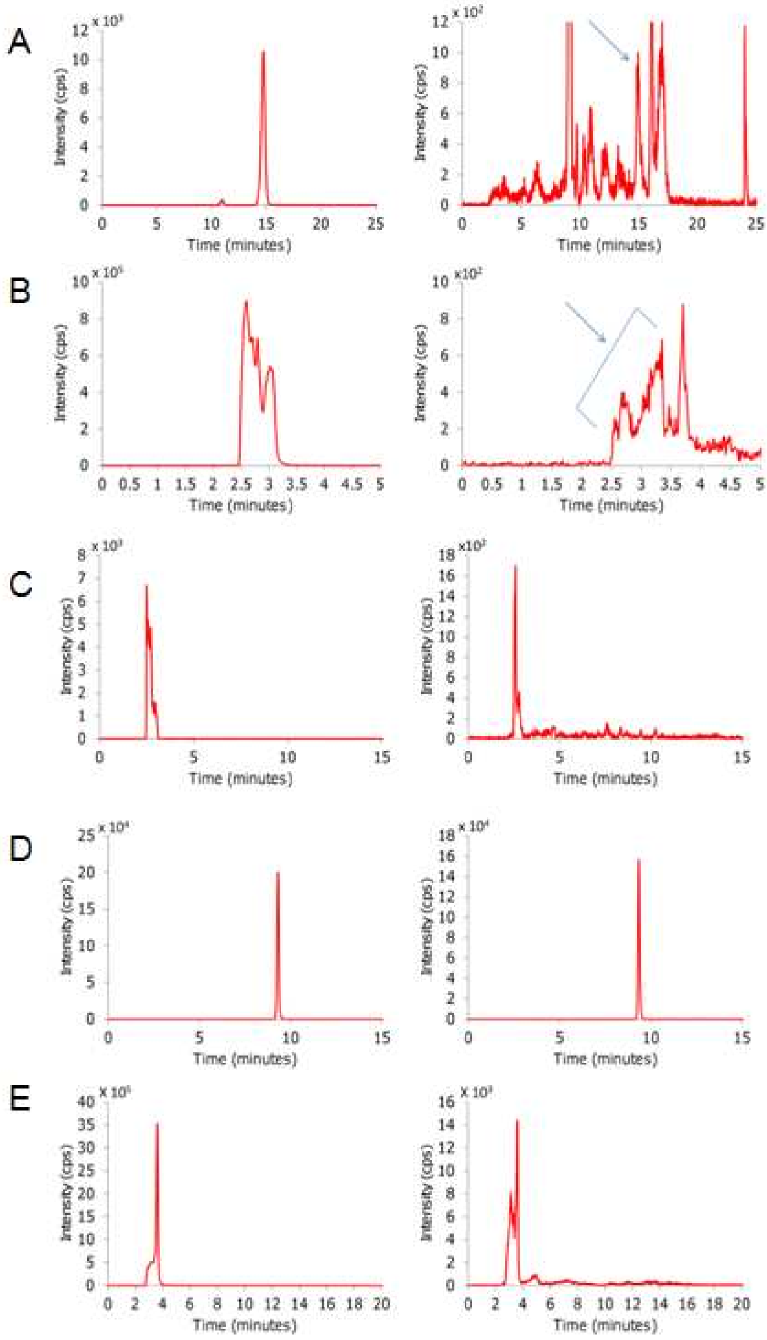


Figure 5-5: Representative chromatograms of all metabolites in both standards and cecum extracts.

Representative chromatograms for standards (left) and metabolite extracts of week 5 cecum samples (right) for (A) Indole 3-Acetate, MRM: 176>130 (B) Indole 3-Acetamide, MRM: 175>130 (C) Hydroxyindole MRM: 134.1>77.1 (D) Tryptophan MRM: 205>188 (E) Tryptamine, MRM: 161>144.2

To determine whether the MRM method could be used to detect physiological differences in metabolite levels, we compared the fecal pellet and cecum concentrations of TRP and its derivatives in young (age 6 weeks) and old (age 15 weeks) female C56BL/6 mice fed standard chow (Figure 5-6).

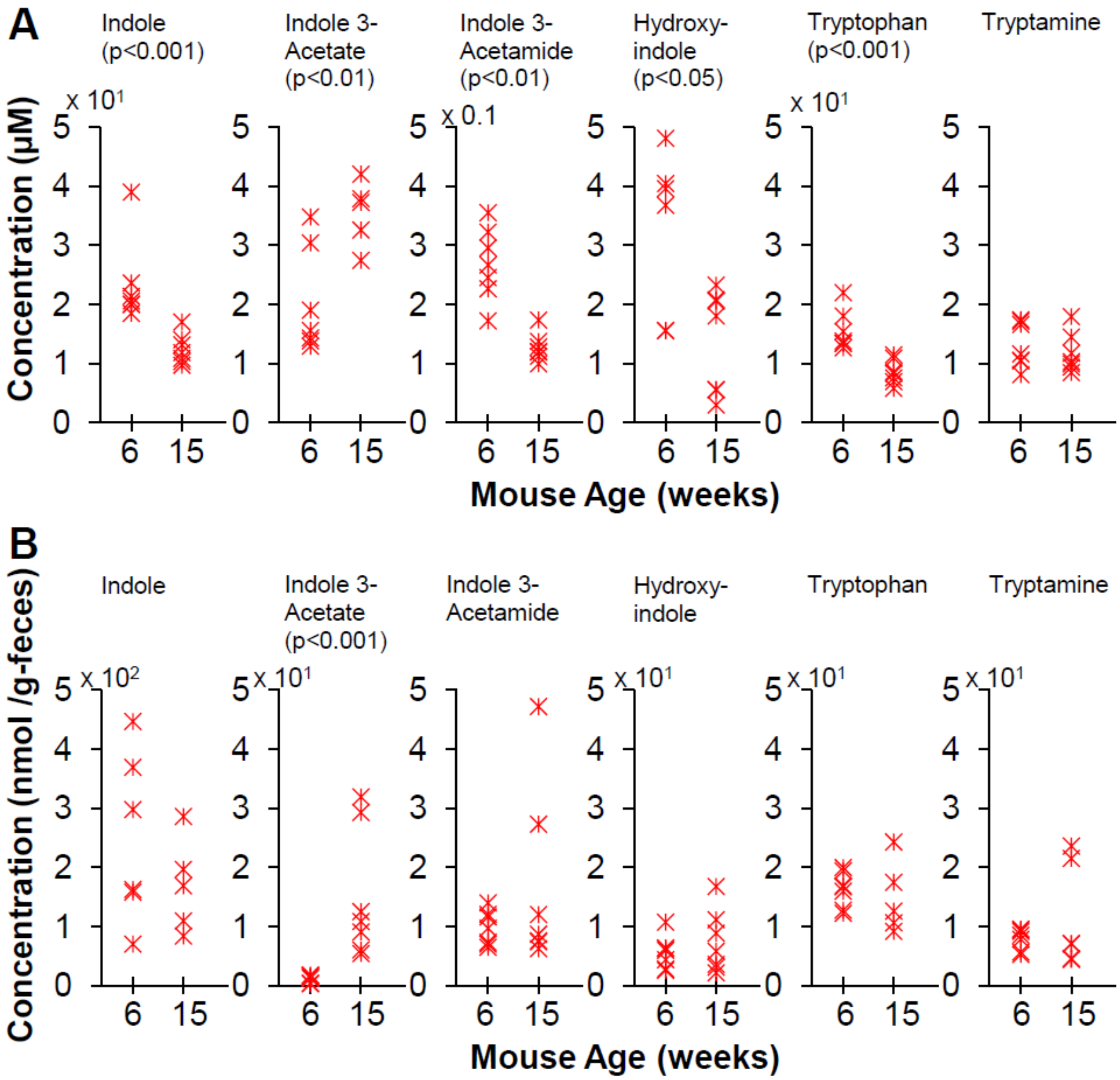


Figure 5-6: ***In vivo*** **concentrations of metabolites in cecum and feces**

All six metabolites were detected in both fecal pellet and cecum extracts from young and old mice. For cecum extracts, statistically significant differences ($p < 0.05$) were found between young and old mice for all metabolites except tryptamine. For fecal extracts, statistically significant differences were found only for indole 3-acetate. Unlike the fecal extracts, metabolite levels in the cecum extracts could be meaningfully expressed as tissue concentrations by normalizing the absolute molar quantities with respect to the weight of the source tissues and approximating tissue density with that of water (1 g/ml). For example, after accounting for the partition of indole between the polar and nonpolar phases of the metabolite extraction solvent (*Materials and Methods*), the normalized tissue concentrations for indole ranged from 9.7 to 40 μM These are conservative estimates, as it is expected that the efficiency for the extraction of metabolites from tissue into solvent is less than 100 %.

5.4.3 Activation of AhR by Microbiota Metabolites

We investigated if the identified microbiota metabolites could be putative ligands in host cells and activate eukaryotic signaling pathways. Specifically, we investigated the ability of identified microbiota metabolites to act as agonists for the AhR, as previous studies (Heath-Pagliuso *et al.*, 1998) have shown that endogenous TRP-derived metabolites can activate AhR signaling. MCF-7 cells with a stably integrated *Gaussia luciferase* (Gluc) reporter plasmid for AhR binding activity were incubated with 100 uM microbiota metabolites or 20 nM TCDD (positive control) for 48 h, and luciferase activity in culture supernatants was measured. Exposure to TCDD, a high-affinity AhR agonist, resulted in a 2-fold increase in the rate of AhR-driven luciferase activity (RLU/h/RFU) as compared to the solvent control (Unpublished Result, Jayaraman Laboratory, Figure 5-7).

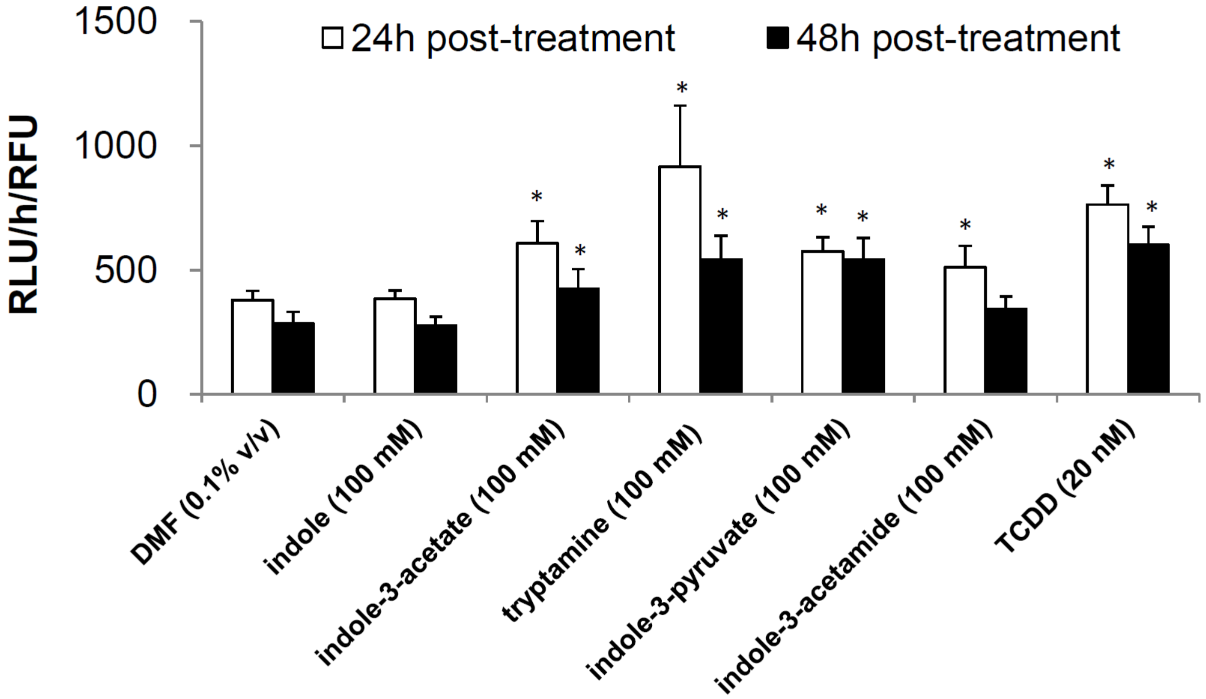


Figure 5-7: Activation of AhR

Exposure to indole-3-acetate, tryptamine and indole-3-pyruvate also resulted in similar induction of AhR activity. Indole-3-acetamide resulted in a 1.3-fold statistically significant (*p* < 0.02) increase in AhR binding activity at 24 h but not at 48 h. Interestingly, indole did not induce activation of AhR, suggesting that the core indole moiety is by itself not sufficient to elicit a significant response.

5.5 Discussion

In this study, we present a methodology for the prediction, identification and quantification of gut microbiota metabolites that integrates computational pathway analysis into a targeted metabolomics workflow. The effects of physiological or pathological perturbations on the microbiota have been investigated using metagenomic analyses characterizing the composition of the gut microbial community, the enrichment (or depletion) of bacterial genes, or the expression levels of genes for specific metabolic pathways (Jones *et al.*, 2008; Handelsman, 2004; Turnbaugh *et al.*, 2006). A limitation of these analyses is that they do not provide direct information on the products of bacterial metabolism such as which molecules are formed from bacterial biotransformation reactions and the concentrations at which these metabolic products are present. Thus, the ability to unambiguously identify bacterial metabolites and quantify their levels in the GI tract is expected to have a significant impact on the study of human gut microbiome function.

One limitation of the *in silico* metabolite prediction step is that its reliability is predicated on the accuracy and completeness of the KEGG pathway database. While the KEGG database is continually and frequently updated, it is certainly possible that our predictions lag behind newly published discoveries regarding microbiota metabolism. For example, Wikoff and coworkers recently identified indole 3-propionate as a TRP-derived metabolite that is produced by the gut microbiota (Wikoff *et al.*, 2009). However, at the time of completion of this work, this metabolite was not listed in the KEGG compound database, and thus could not be identified by our algorithm. Similarly, the discrimination of bacterial metabolites from metabolites that could also be produced by host cells depends on an accurate model of host metabolism. The most relevant host genomes, e.g., mouse and human, have been sequenced and largely annotated, and several published models are available that exhaustively catalogue the metabolic reactions. However, prior studies with genome-scale model reconstruction suggest that several iterations may be required until the published models can be considered stable, consensus reconstructions (Aziz *et al.*, 2012).

Another limitation of the prediction algorithm is that it does not differentiate between reactions based on their gene abundance or expression level when constructing a possible biotransformation route. Consequently, each candidate reaction that can be connected to the source metabolite has an equal likelihood of selection, which unlikely reflects the true engagements of metabolic reactions in the gut microbiota. In principle, metabolites produced by reactions encoded by genes that are highly abundant and/or highly expressed should more likely be present at quantifiable levels compared to the products of depleted or minimally expressed pathways. In this regard, both of the present limitations of the *in silico* prediction step could addressed by incorporating metagenomic data on as they become available. Our pathway construction already accepts user-specified selection weights, and could be extended in a straightforward manner to explore a microbiota metabolic network weighted by relative gene abundances or expression levels.

A key advantage of MRM as a MS technique for targeted metabolomics is that the detection of precursor-product ion pairs offers greater specificity compared to full scan MS, while allowing absolute quantitation. This targeted approach also allows for enhanced sensitivity as well as improved LOD, because instrument-specific parameters can be tailored and optimized for the detection of each individual MRM transition, whereas full scan methods are restricted to one fixed set of parameters for all analytes, which may be suboptimal for the ionization and fragmentation of certain metabolites. In practice, even optimized MRM transitions may not represent a unique mass signature for a metabolite, as there are cases where multiple analytes present in a biological sample share the same transition. For example, indole 3-acetamide shows a strong signal for the 175 → 130 transition, as does arginine, a highly abundant amino acid. This overlap in MRM transitions by different compounds underscores the critical importance of chromatographic separation in identifying metabolites. Another common problem that potentially compromises specificity is the tendency for metabolites possessing thermally labile bonds to decompose due to heated electrospray ionization (HESI). For example, a sample containing only high-purity TRP showed a strong signal for the indole transition (118 → 91) at the same retention time as TRP (Figure 5-4), presumably due to partial decomposition into indole at the ion source.

To our knowledge, this is the first study to use MRM to quantify physiological concentrations of microbiota-produced metabolites present in intestinal tissue extracts. While the literature on absolute concentrations of microbiota metabolites is relatively sparse, we found good agreement between our results and previously reported values. In an early study, Whitt and coworkers used an enzymatic assay to determine an indole concentration of ~40 nmol/g tissue in murine cecum (Whitt and Demoss, 1975), which is comparable to our results (16 - 31 nmol/g tissue in cecum of young mice). The earlier study also reported that indole was absent in the cecum of germ-free mice, which is consistent with our pathway analysis algorithm's prediction that indole is a product of bacterial metabolism. A comparison of samples from young and old mice suggests that the age of the host impacts the levels of microbiota metabolites in the GI tract as reflected in cecum and fecal concentrations of bacterial TRP derivatives. This observation is consistent with several recent studies showing that age may influence levels of other microbiota-associated metabolites. For example, Vaahtovuo and coworkers reported differences in bacteria-derived cellular fatty acids in stool samples between 5 - 7 and 15 - 19 weeks old mice (Vaahtovuo *et al.*, 2001).

The AhR is a ligand-activated transcription factor that plays an important role in the mucosal immune system (Ying Li *et al.*, 2011), and several TRP-derived chemicals have been identified as AhR ligands (Bjeldanes *et al.*, 1991). In our study, we observed that three of the predicted TRP derivatives (indole 3-acetate,tryptamine and indole 3-pyruvate) were able to activate AhR in a dose-dependent manner (Figure 5-7). This result is consistent with a previous study by Heath-Pagliuso *et al.*, who showed that tryptamine and indole-3-acetate function as AhR agonists (Heath-Pagliuso *et al.*, 1998). However, this previous study, while assuming that these two metabolites could be derived from TRP through enzymatic reactions, did not confirm that they are actually present in the GI tract in quantifiable amounts. In this work, we show that putative AhR agonists like tryptamine and indole-3-acetate are present in the cecum at *intracellular concentrations* (i.e., the levels present in the cecal bacteria and the surrounding tissue) ranging from 1 to 5 µM, and that they activate the AhR *in vitro* at concentrations of ~100 µM. This apparent discrepancy between the measured concentration of metabolites and the concentrations required to activate the AhR *in vitro* could be due to two reasons. First, the measured concentration of these metabolites is a conservative estimate, as it does not account for the extraction efficiency. Second, the concentrations needed to activate the AhR *in vitro* are *extracellular concentrations* (i.e., concentrations in the culture medium), and since the AhR is an intracellular (cytosolic) receptor, the extracellular medium concentrations may not equate to intracellular levels, depending on the rates of uptake and metabolism inside the cells. Further analysis is needed to model the kinetics of ligand uptake and processing that lead to the activation of the AhR.

An obvious extension of this work is to predict and identify molecules that can be derived by the intestinal microbiota from different source metabolites. One such source metabolite could be the amino acid phenylalanine that is reduced in the serum of germ-free mice, and therefore, a source of putative bioactive molecules. The methodology described here can also be applied to identifying bioactive derivatives of environmental contaminants such as bisphenol A that can be generated *in vivo* by the GI tract microbiota and either have increased activity or have potentially different spectrum of activity (Van de Wiele *et al.*, 2005).

6 Future Directions and Recommendations

We aim to demonstrate in this thesis that modularity analysis of metabolic networks enables one to study the functional organization of cellular metabolism in ways that many not be immediately intuitive looking at a two-dimensional atlas. Modularity serves to identify groups of reactions that may mutually influence each others' reaction kinetics, and as a whole module, possesses structural features that confer robustness. In the first chapter, we introduced both global and local robustness, and showed that cyclical interactions are important structural motifs that provide those properties. Using ShReD-based partitioning on either un-weighted or metabolic flux-based weighted reaction-centric networks, one can uncover functional modules that are enriched in cyclical interactions. However, whether or not ShReD-based modules actually do capture a subset of reactions that isolate external perturbations from propagating to the rest of the network still remains a challenging research question yet to be addressed.

One approach could be to use pseudo-kinetic models guided by experimental metabolomics data and apply metabolic control analysis to show that for a particular perturbed enzyme, flux control coefficients for fluxes within the perturbed module are on average greater than flux control coefficients for fluxes outside the perturbed module. In fact, the formalism for modular control analysis has been recently proposed (Acerenza and Ortega, 2007; Ortega and Acerenza, 2011), as has control analysis in the context of metabolic robustness (Quinton-Tulloch *et al.*, 2013b). However, the main challenge with this approach is that the features identified by ShReD require detailed topology of larger scale networks, but kinetic models are only available for simpler collapsed networks. For example, Klaus and coworkers provide a kinetic model for the hepatocyte using linlog kinetics (Visser, 2003; Heijnen, 2005) based on transient metabolomics data on intracellular metabolite concentrations and also provide flux control coefficients for each enzyme-flux pair (Maier *et al.*, 2010). However, their model stoichiometry is simplified to only 69 reactions, compared to the 2539 reactions in the large scale hepatocyte model (hepatonet1) presented by Gille and coworkers (Gille *et al.*, 2010). As a result, while we have some information on the dynamics of the hepatocyte, relating structural partitioning identified using graph-based modularity analysis to flux control is limited because of the topological information lost in the model reduction process.

Moreover, even the structural analysis of detailed large scale networks faces its own set of limitations. Many models are still solely based on stoichiometry and do not have embedded regulatory information. The Brenda enzyme database (I. Schomburg *et al.*, 2002) provides citations to publications referring to allosteric regulation for each documented enzyme involved in metabolism. However, incorporating this information into tissue or organ specific models not an easily automatable process because *ad hoc* judgments are sometimes required regarding whether or not a particular allosteric regulation mechanism applies cross-species. For example, much of the literature on allosteric regulatory interactions has involved bacterial species. While the structure and function of many enzymes have been evolutionarily conserved, it is still difficult to determine if regulatory mechanisms identified in bacterial systems applies to higher order organisms. It is also challenging to determine the relative strength of regulatory interactions. Indeed, known Km saturation constants along with absolute metabolite concentration levels using metabolomics approaches should provide some information on how to appropriately weight the regulatory links.

With biological research, decisive experimentation is what ultimately provides tangible and beneficial outcomes. Modularity analysis serves as a modeling platform to provide novel representations of metabolism and help predict cellular responses to various experimental metabolic interventions. As discussed, the aforementioned methodologies have great room for improvement. However, with ever expanding databases consisting of tissue-specific gene expression data, regulatory information, as well as a recent surge in establishing high throughput metabolomics protocols (e.g. monitoring intracellular redox, Appendix B) should all aid in bridging the gap between theoretical systems biology and experimental biochemistry.

Appendix A: Supplementary for Chapter 4

Table S1: Adipocyte model reaction definitions

Reaction	Stoichiometry
	Transport
1	Glucose-E → Glucose
2	Lactate → Lactate-E
3	Glycerol → Glycerol-E
4	Palmitate → Palmitate-E
5	TG → TG-E
6	Alanine-E → Alanine
7	Aspartate-E → Aspartate
8	Asparagine-E → Asparagine
9	Glutamine-E → Glutamine
10	Glutamate-E → Glutamate
11	Glycine-E → Glycine
12	Proline-E → Proline
13	Serine-E → Serine
14	Tyrosine-E → Tyrosine
15	Histidine-E → Histidine
16	Isoleucine-E → Isoleucine
17	Leucine-E → Leucine
18	Lysine-E → Lysine
19	Methionine-E → Methionine
20	Phenylalanine-E → Phenylalanine
21	Threonine-E → Threonine
22	Valine-E → Valine
23	O_2-E → O_2
	Carbohydrate Metabolism
24	Glucose + ATP → Glucose 6-Phosphate + ADP
25	Glucose 6-Phosphate → Fructose 6-Phosphate
26	Fructose 6-Phosphate + ATP → Glyceraldehyde 3-Phosphate + ADP
27	Glycerone-Phosphate → Glyceraldehyde 3-Phosphate
28	Glyceraldehyde 3-Phosphate + NAD + ADP → 3-Phosphoglycerate + NADH + ATP
29	3-Phosphoglycerate → Phosphoenolpyruvate
30	Phosphoenolpyruvate + ATP → Pyruvate + ADP
31	Pyruvate + NADH → Lactate + NAD
32	Glucose 6-Phosphate + 2 NADP → 2 NADPH + Ribulose 5-Phosphate
33	Ribulose 5-Phosphate + Erythrose 4-Phosphate → Fructose 6-Phosphate + Glyceraldehyde 3-Phosphate
	TCA Cycle/ Oxidative Phosphorylation
34	Pyruvate-M + Oxaloacetate-M + NAD-M → NADH-M + Citrate-M
35	Pyruvate-M + ATP → Oxaloacetate-M + ADP
36	Citrate-M + NAD-M → 2-Oxoglutarate-M + NADH-M
37	2-Oxoglutarate-M + CoA + NAD-M → NADH-M + Succinyl-CoA-M

38	Succinyl-CoA-M + ADP → CoA + Succinate-M + ATP
39	FAD-M + Succinate-M → FADH$_2$-M + Fumarate-M
40	Fumarate-M → Malate-M
41	Malate-M + NAD-M → NADH-M + Oxaloacetate-M
48	NADH-M + 0.5 O$_2$ + 2.3 ADP → 2.3 ATP + NAD-M
49	FADH$_2$-M + 0.5 O$_2$ + 1.4 ADP → 1.4 ATP + FAD-M
	Lipid Metabolism
46	Oxaloacetate + ATP → Phosphoenolpyruvate + ADP
47	Phosphoenolpyruvate + 2 ATP + 2 NADH → Glycerone-Phosphate + 2 NAD + 2 ADP
50	8 Acetyl-CoA + 14 NADPH + 7 ATP → 8 CoA + 14 NADP + Palmitate + 7 ADP
51	Glycerone-Phosphate + 3 Palmitate + NADH → NAD + Tripalmitoylglycerol
52	Tripalmitoylglycerol → 3 Palmitate + Glycerol
	Citrate/ Malate Shuttle
42	Citrate + CoA + ATP → Acetyl-CoA + Oxaloacetate + ADP
43	Oxaloacetate + NADH → Malate + NAD
44	Malate + NADP → NADPH + Pyruvate
45	Citrate + NADP → 2-Oxoglutarate + NADPH
53	Pyruvate → Pyruvate-M
54	Citrate-M → Citrate
55	2-Oxoglutarate + Malate-M → 2-Oxoglutarate-M + Malate
	Amino Acid Metabolism
56	Pyruvate + Glutamate → 2-Oxoglutarate + Alanine
57	ATP + Aspartate + Glutamine → Glutamate + Asparagine
58	Oxaloacetate + Glutamate → 2-Oxoglutarate + Aspartate
59	Glutamine → Glutamate
60	2-Oxoglutarate + NADH → NAD + Glutamate
61	Serine → Glycine
62	Serine → Pyruvate
63	2 NADH + Glutamate → 2 NAD + Proline
64	Phenylalanine → Tyrosine
65	Tyrosine → Fumarate-M
66	Histidine → Glutamate
67	Isoleucine + 2 CoA → FADH$_2$-M + Acetyl-CoA + 2 NADH-M + Succinyl-CoA-M
68	Leucine + ATP → Acetyl-CoA + FADH$_2$-M + 2 NADH-M
69	Lysine + CoA → FADH$_2$-M + 3 NADH + 2 NADH-M
70	Serine + Methionine + ATP + CoA → Acetyl-CoA + NADH
71	Threonine + CoA → Acetyl-CoA + NADH + Glycine
72	Valine + CoA → FADH$_2$-M + 4 NADH + Succinyl-CoA-M

Suffixes E and M refer to extracellular and mitochondrial metabolites, respectively.

Table S2: Flux distributions

Reaction	Day 4	Day 12	Day 12 w/ LDH inhibition	Day 12 w/ PCX inhibition
1	268.0	266.0	212.0	341.0
2	359.7	200.0	119.0	428.0
3	5.0	8.6	19.8	20.6
4	0.3	1.1	1.0	0.8
5	0.9	4.8	2.3	2.9
6	-15.4	0.7	0.6	-0.5
7	0.0	0.0	0.0	0.0
8	-2.4	-3.0	-1.1	-4.3
9	4.3	-11.2	-12.0	4.3
10	1.5	0.4	0.4	0.3
11	-8.8	-7.2	-5.3	-6.8
12	-1.2	-0.4	-0.5	-2.1
13	9.2	5.7	4.6	5.7
14	0.6	-0.4	0.0	0.0
15	2.5	0.0	0.0	0.0
16	9.5	8.2	6.1	7.5
17	10.6	8.8	6.8	7.9
18	2.8	0.3	0.0	0.0
19	0.8	0.0	0.0	0.0
20	0.6	0.4	0.0	0.0
21	0.8	1.6	0.6	1.1
22	5.8	5.5	3.3	3.8
23	499.0	726.2	662.7	617.0
24	268.0	266.0	212.0	341.0
25	265.8	209.1	186.6	300.2
26	268.0	266.0	212.0	341.0
27	472.3	628.0	483.4	580.9
28	742.5	950.9	720.9	962.7
29	742.5	950.9	720.9	962.7
30	532.3	655.6	462.6	736.6
31	359.7	200.0	119.0	428.0
32	2.2	56.9	25.4	40.8
33	2.2	56.9	25.4	40.8
34	178.8	375.7	305.1	280.3
35	1.1	137.8	69.6	63.7
36	161.2	224.2	226.2	205.4
37	183.6	259.0	243.3	223.3
38	198.9	272.7	252.7	234.6
39	198.9	272.7	252.7	234.6
40	200.1	272.7	252.7	234.6
41	177.7	237.9	235.6	216.7
42	2.4	105.4	49.7	59.5

43	0.0	22.3	13.3	18.0
44	22.4	57.2	30.5	35.9
45	15.3	46.1	29.3	15.5
46	0.0	80.1	35.3	37.3
47	210.2	375.4	293.5	262.4
48	762.6	1148.3	1049.6	972.3
49	235.3	304.0	275.7	261.7
50	3.0	15.5	7.9	9.5
51	5.9	13.4	22.1	23.5
52	5.0	8.5	19.8	20.6
53	180.0	513.5	374.7	344.0
54	17.6	151.5	79.0	75.0
55	22.4	34.9	17.2	17.9
56	15.4	-0.7	-0.6	0.5
57	2.4	3.0	1.1	4.3
58	2.4	3.0	1.1	4.3
59	1.9	14.2	13.1	0.0
60	10.7	13.5	12.6	2.3
61	8.0	5.7	4.6	5.7
62	0.4	0.0	0.0	0.0
63	1.2	0.4	0.5	2.1
64	0.6	0.4	0.0	0.0
65	1.2	0.0	0.0	0.0
66	2.5	0.0	0.0	0.0
67	9.5	8.2	6.1	7.5
68	10.6	8.8	6.8	7.9
69	10.6	8.8	6.8	7.9
70	0.8	0.0	0.0	0.0
71	0.8	1.6	0.6	1.1
72	5.8	5.5	3.3	3.8

All units are in mmol/g-DNA/2 days. Values shown are mean fluxes (n = 6). Fluxes for the Day 4 condition were calculated using data taken from (Si et al, 2007). Fluxes for all other conditions were calculated using data taken from (Si et al, 2009).

Appendix B: LC/MS-MS Method for Quantifying Cofactor Ratios and Correlation with Optical Fluorescence Microscopy

Abstract/Summary

The non-invasive spatial mapping of cell metabolic status within a tissue could provide substantial advancements in disease diagnosis, assessing therapeutic efficacy, and understanding tissue development. Here, we describe a two-photon excited fluorescence microscopy technique using only endogenous cell fluorescence to assess the metabolic status of mesenchymal stem cells undergoing adipogenic and osteoblastic differentiation. Using liquid chromatography/ mass spectrometry and quantitative PCR, we validate the sensitivity of an optical redox ratio of FAD/(NADH+FAD) to stem cell differentiation. Furthermore, we probe the underlying physiological mechanisms, which relate the onset of differentiation to a decrease in the optical redox ratio. Because traditional assessments of stem cells and engineered tissues are destructive, time consuming, and logistically intensive, the development and validation of a non-invasive, label-free approach to defining the spatiotemporal patterns of cell differentiation can offer a powerful tool for rapid, high-content characterization of cell and tissue cultures.

LC/MS-MS Correlations with Optical Redox Ratio

To validate our optical sensitivity to the cofactors NADH and FAD, intracellular metabolites were extracted and measured through liquid chromatography / mass spectrometry (LC/MS-MS).

In order to achieve sufficient signal intensity from the mass spectrometer (Figure Appendix B-1) using a single cell culture, confluent T75 flasks of differentiating MSCs were used.

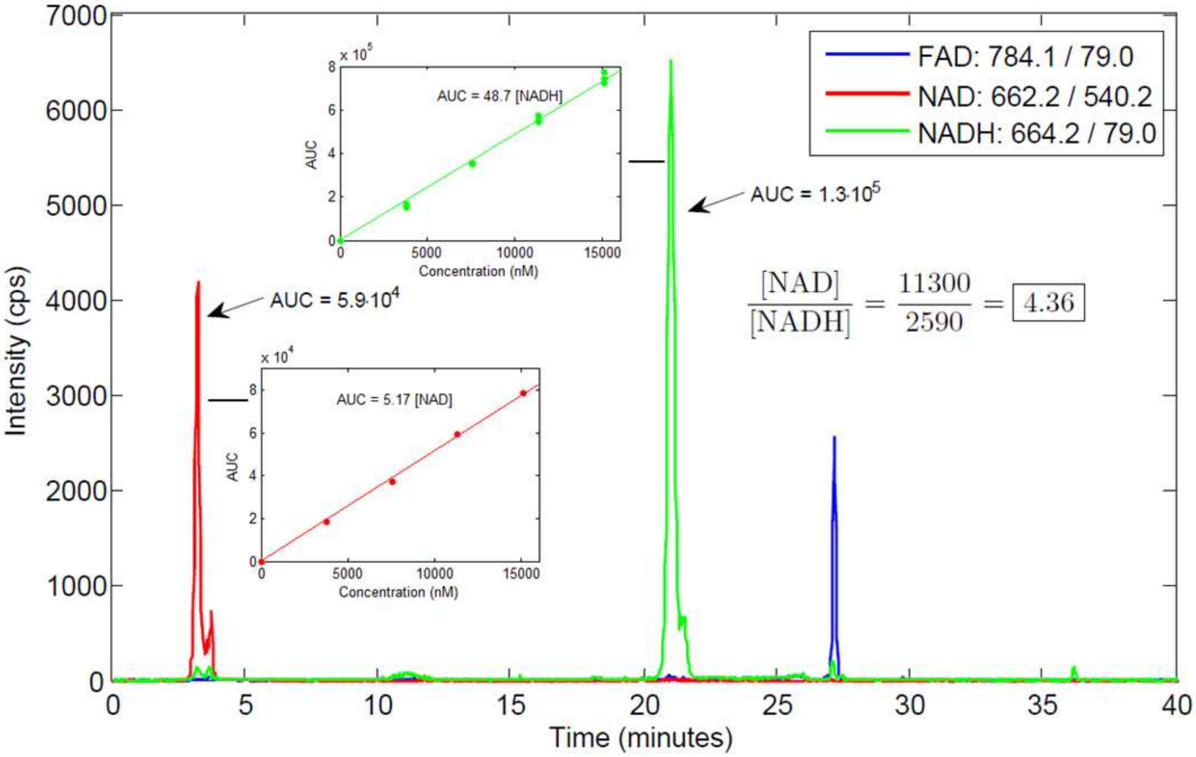


Figure Appendix B-1: Standards of NAD, NADH, and FAD

To facilitate optical imaging of each T75 flask with our high NA objective, a 3/8 inch hole was drilled in a sterile environment and a 25mm glass coverslip (No 1.5) was adhered to the flask using an aquarium-safe silicone sealant. MSC cultures undergoing adipogenic differentiation, osteogenic differentiation, or MSC propagation at a range of post-induction time points up to 2 weeks (n=22 cultures) were assessed through TPEF imaging (n=10 fields per culture) and immediately sacrificed for metabolite extraction.

To extract intracellular metabolites from each cell culture after imaging, a previously established extraction protocol was modified. Briefly, cells were trypsinized and centrifuged to form pellets, which were collected in 1.5mL microfuge tubes. A 2:1 mixture of methanol:chloroform by volume (500µL) was added to each pellet. The pellets were flash frozen in liquid nitrogen for 30 seconds and then allowed to thaw, followed by a brief 10 second vortex. This freeze/thaw step was then repeated twice, and then 200µL of ice cold water was added to each sample. Samples were then centrifuged for 1 minute at 15,000 g to form a biphasic mixture, and the upper (polar) phase was collected for LC/MS-MS analysis.

The LC/MS-MS analysis was performed using a 3200 QTRAP Hybrid Triple Quadrupole Linear Ion Trap mass spectrometer (AB SCIEX, Foster City, CA) coupled to a 1200 Series Binary LC System (Agilent Technologies, Santa Clara, CA). To establish standard curves for NAD^+, NADH, and FAD, 0.01g of each compound was dissolved in 10mL of 50:50 v% methanol:water to obtain a 1g/L stock solution. The stock solutions were diluted 100-fold, and subsequently serially diluted to obtain a set of standards ranging from 1-10mg/L. All compounds and solvents were purchased from Sigma-Aldrich (St. Louis, MO).

The chromatographic separation, slightly modified from a previous protocol, was achieved using a Synergi Fusion-RP (reversed phase) column (150 mm X 2 mm inner diameter, 4 µm 80Å particles; Phenomenex, Torrance, CA), which was kept at ambient temperature. The aqueous mobile phase A was composed of 15mM tributyl amine and 10mM acetic acid in HPLC grade water. The solution was stirred overnight as the pH gradually equilibrated to 8.45. Mobile phase B was pure methanol. The separation was performed using an LC gradient that was determined to provide the desired peak quality. Injection volume for each sample was 10µL.

Appendix B, Table 1

Step	**Time (min)**	**Flow rate (μL/min)**	**A (%)**	**B (%)**
1	0	200	80	20
2	5	200	80	20
3	10	200	68	32
4	15	200	65	35
5	25	200	40	60
6	50	200	10	90
7	55.1	200	10	90
8	55.2	200	80	20
9	62	200	80	20

Mobile phase A = 15mM tributyl amide and 10mM acetic acid

Mobile phase B = methanol

To measure each compound, a 10mg/L solution was injected using a syringe via direct infusion at 10μL/min. The mass spectrometer was operated in negative mode with the curtain gas (CUR) set at 30.0, collision gas (CAD) at 5, IonSpray Voltage (IS) at -4500.0, the temperature (TEM) of the turbo gas in the TurboIonSpray at 400°C, and both Ion Source Gases (GS1 and GS2) at 60.0. Several compound dependent parameters, including the declustering potential (DP), the entrance potential (EP), the collision energy (CE), as well as the collision cell exit potential (CXP) were selected to optimize the intensity of the precursor/product transition for each analyte (Appendix B, Table 2).

Appendex B, Table 2

Compound	**Precursor (Da)**	**Product (Da)**	**DP (V)**	**EP (V)**	**CE (V)**	**CXP (V)**
NAD^+	662.2	540.2	-10	-10.5	-22	-22
NADH	664.2	79	-65	-4.5	-128	-4
FAD	784.1	79	-55	-9	-130	-2

The mass spectrometer acquisition method was operated in multiple reaction monitoring (MRM) mode, simultaneously scanning for the precursor/product transitions of NAD^+, NADH, and FAD. The dwell time for each analyte was 500ms, and the total cycle time was therefore 1.5s with 2455 cycles per sample run.

For both the standards and cell extracts, the areas under the NAD^+, NADH, and FAD peaks were quantified using Analyst software (Version 1.5, AB Sciex). (Standard curves exceeding R^2=0.995 were achieved, and used to calculate the concentration of each cofactor. Based on the measured concentrations, redox ratios of FAD/(NADH+FAD) and NAD^+/(NADH+NAD^+) were computed for each culture, and a linear regression with the optical redox ratio was performed.

Results

To confirm that this optical redox ratio is both sensitive *and* specific to intracellular changes in NADH and FAD concentrations within differentiating stem cell cultures, the average optical redox ratio was compared with the ratio of cofactor concentrations extracted and measured using liquid chromatography / tandem mass spectrometry (LC/MS-MS). Bone marrow-derived mesenchymal stem cell (MSC) cultures (n=22) undergoing adipogenic differentiation, osteogenic differentiation, or MSC propagation at different post-induction time points were monitored through TPEF imaging and immediately sacrificed for metabolite extraction. The average optical redox ratio of each culture prior to metabolite extraction exhibited a strong correlation (R=0.765, p<0.0001) with the intracellular ratio of NAD$^+$/(NADH+NAD$^+$) regardless of the treatment group, suggesting FAD fluorescence is in direct equilibrium with local NAD$^+$/NADH ratios (Figure Appendix B-2).

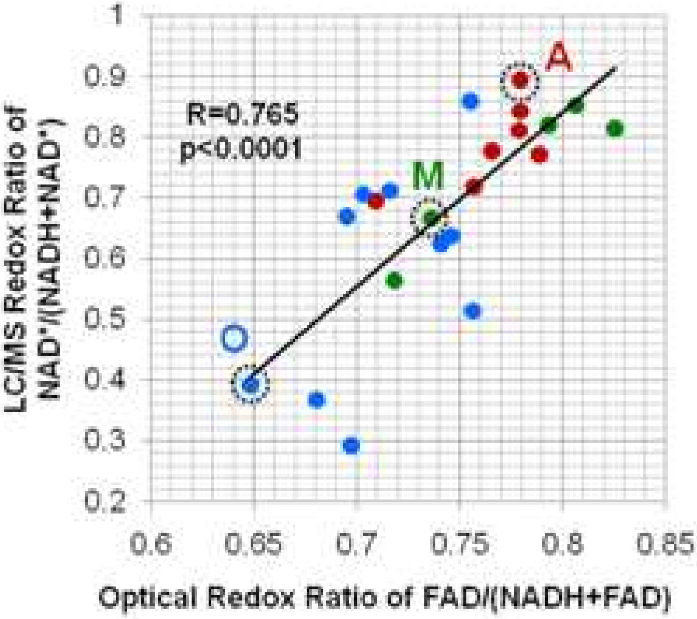


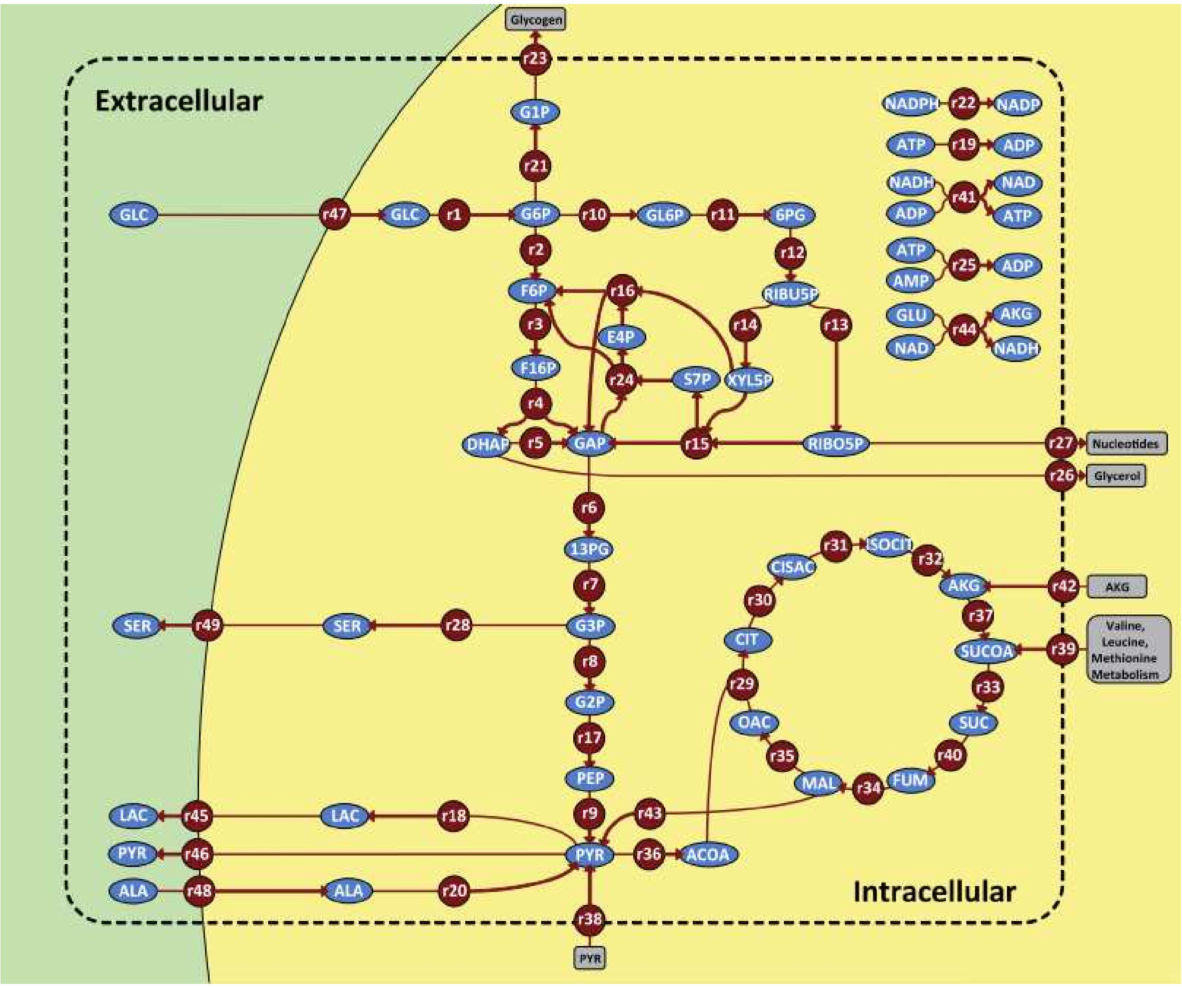
Figure Appendix B-2: Correlation between LC/MS-MS measurement of redox and optical redox ratio

Appendix C: Source Code and Example Model

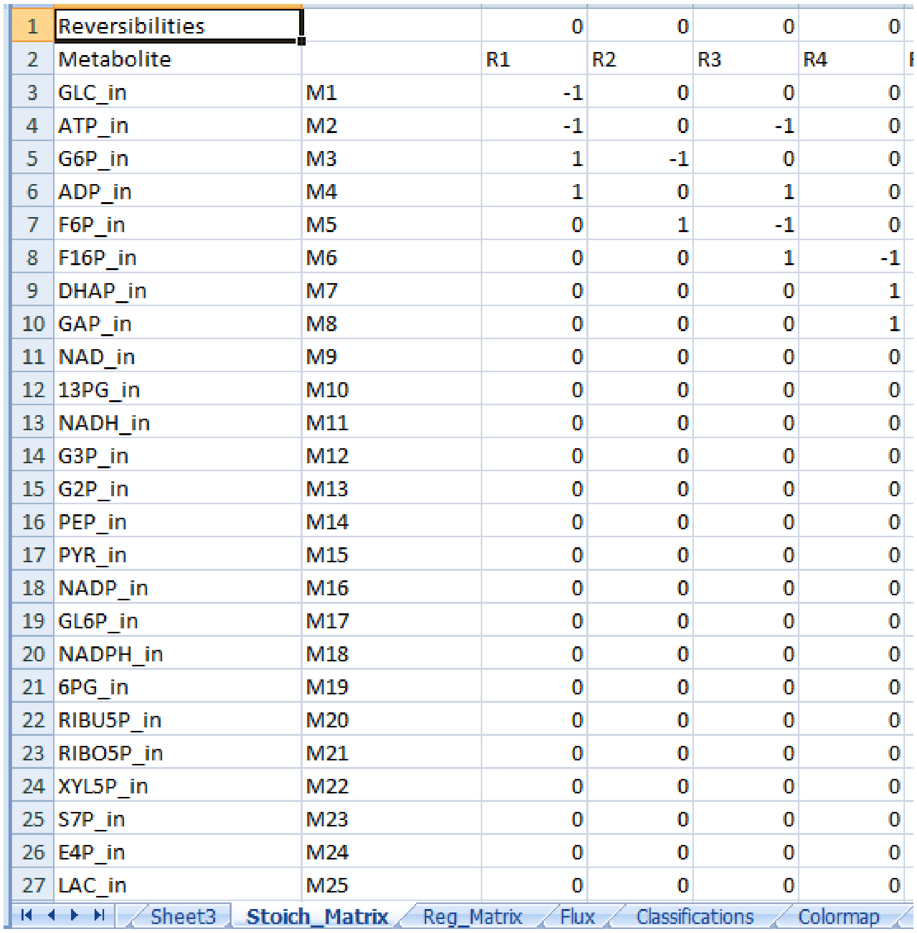
In this section, all the source code for the ShReD-based partitioning of a metabolic model with fluxes is provided in this section. This includes the code for constructing the hierarchical tree of partitions

Model Format

The input to the Shred_Network() function is an excel file for the metabolic model. The proper format for the model is provided below for a liver metabolic network provided by Maier and coworkers (Maier *et al.*, 2010).

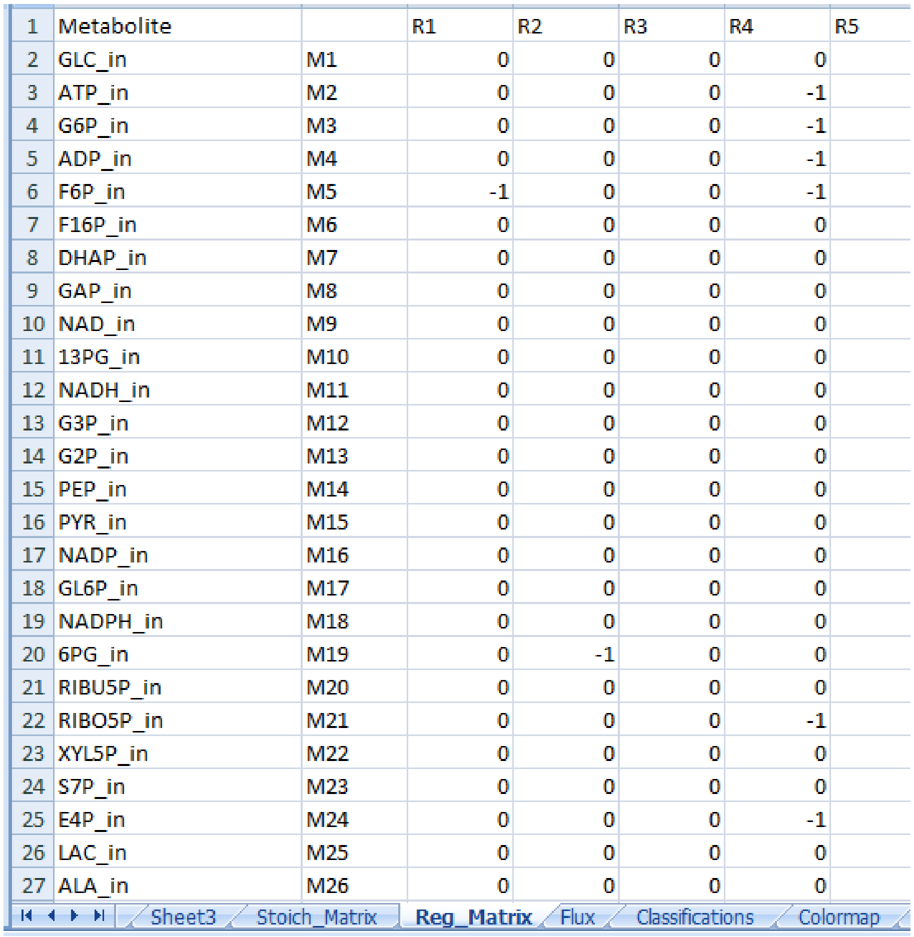


Stoich_Matrix: This sheet contains the stoichiometric matrix of the model along with reversibilities. Irreversible reactions are denoted by 0 and reversible reactions by 1. Only metabolites that should mediate interactions between reactions are included. For example, cofactors such as NAD and NADPH are included. However, hub metabolites such as H_2O or protons (H^+) are not included.



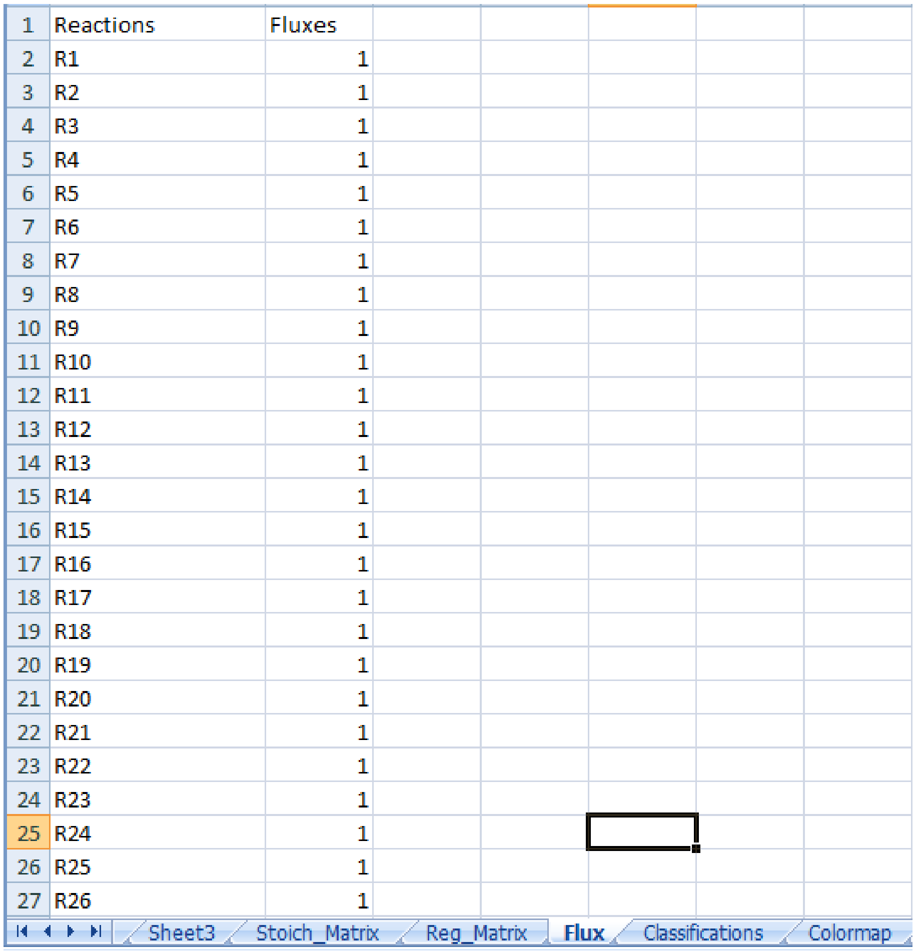
1	Reversibilities		0	0	0	0
2	Metabolite		R1	R2	R3	R4
3	GLC_in	M1	-1	0	0	0
4	ATP_in	M2	-1	0	-1	0
5	G6P_in	M3	1	-1	0	0
6	ADP_in	M4	1	0	1	0
7	F6P_in	M5	0	1	-1	0
8	F16P_in	M6	0	0	1	-1
9	DHAP_in	M7	0	0	0	1
10	GAP_in	M8	0	0	0	1
11	NAD_in	M9	0	0	0	0
12	13PG_in	M10	0	0	0	0
13	NADH_in	M11	0	0	0	0
14	G3P_in	M12	0	0	0	0
15	G2P_in	M13	0	0	0	0
16	PEP_in	M14	0	0	0	0
17	PYR_in	M15	0	0	0	0
18	NADP_in	M16	0	0	0	0
19	GL6P_in	M17	0	0	0	0
20	NADPH_in	M18	0	0	0	0
21	6PG_in	M19	0	0	0	0
22	RIBU5P_in	M20	0	0	0	0
23	RIBO5P_in	M21	0	0	0	0
24	XYL5P_in	M22	0	0	0	0
25	S7P_in	M23	0	0	0	0
26	E4P_in	M24	0	0	0	0
27	LAC_in	M25	0	0	0	0

Reg_Matrix: If regulatory information is to be included for the ShReD partitioning, the regulatory matrix should hold a +1 if the metabolite is a positive regulator of a reaction and a -1 if the metabolite negatively inhibits the reaction.



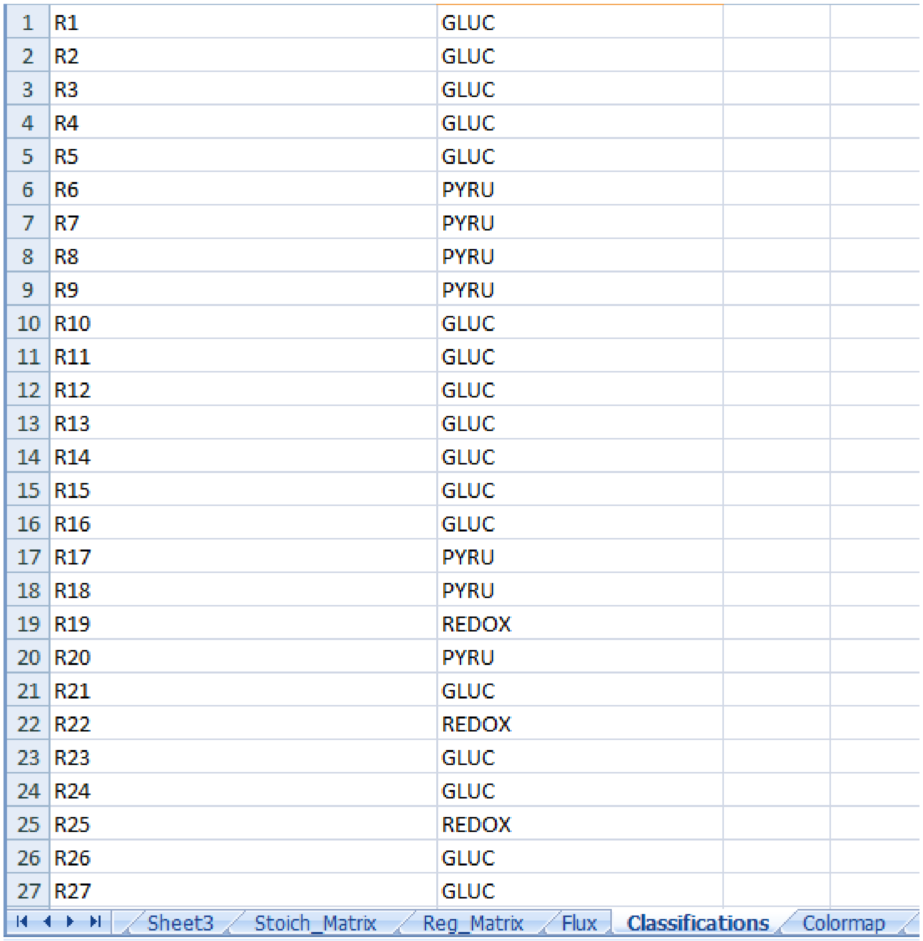
1	Metabolite		R1	R2	R3	R4	R5
2	GLC_in	M1	0	0	0	0	
3	ATP_in	M2	0	0	0	-1	
4	G6P_in	M3	0	0	0	-1	
5	ADP_in	M4	0	0	0	-1	
6	F6P_in	M5	-1	0	0	-1	
7	F16P_in	M6	0	0	0	0	
8	DHAP_in	M7	0	0	0	0	
9	GAP_in	M8	0	0	0	0	
10	NAD_in	M9	0	0	0	0	
11	13PG_in	M10	0	0	0	0	
12	NADH_in	M11	0	0	0	0	
13	G3P_in	M12	0	0	0	0	
14	G2P_in	M13	0	0	0	0	
15	PEP_in	M14	0	0	0	0	
16	PYR_in	M15	0	0	0	0	
17	NADP_in	M16	0	0	0	0	
18	GL6P_in	M17	0	0	0	0	
19	NADPH_in	M18	0	0	0	0	
20	6PG_in	M19	0	-1	0	0	
21	RIBU5P_in	M20	0	0	0	0	
22	RIBO5P_in	M21	0	0	0	-1	
23	XYL5P_in	M22	0	0	0	0	
24	S7P_in	M23	0	0	0	0	
25	E4P_in	M24	0	0	0	-1	
26	LAC_in	M25	0	0	0	0	
27	ALA_in	M26	0	0	0	0	

Flux: This sheet lists the metabolic flux for each reaction in the model. In this particular example, the ShReD-based partitioning was done for the un-weighted case so all flux values are listed as 1. However, if flux-based weights are desired, then the metabolic flux distribution should be listed in this sheet.



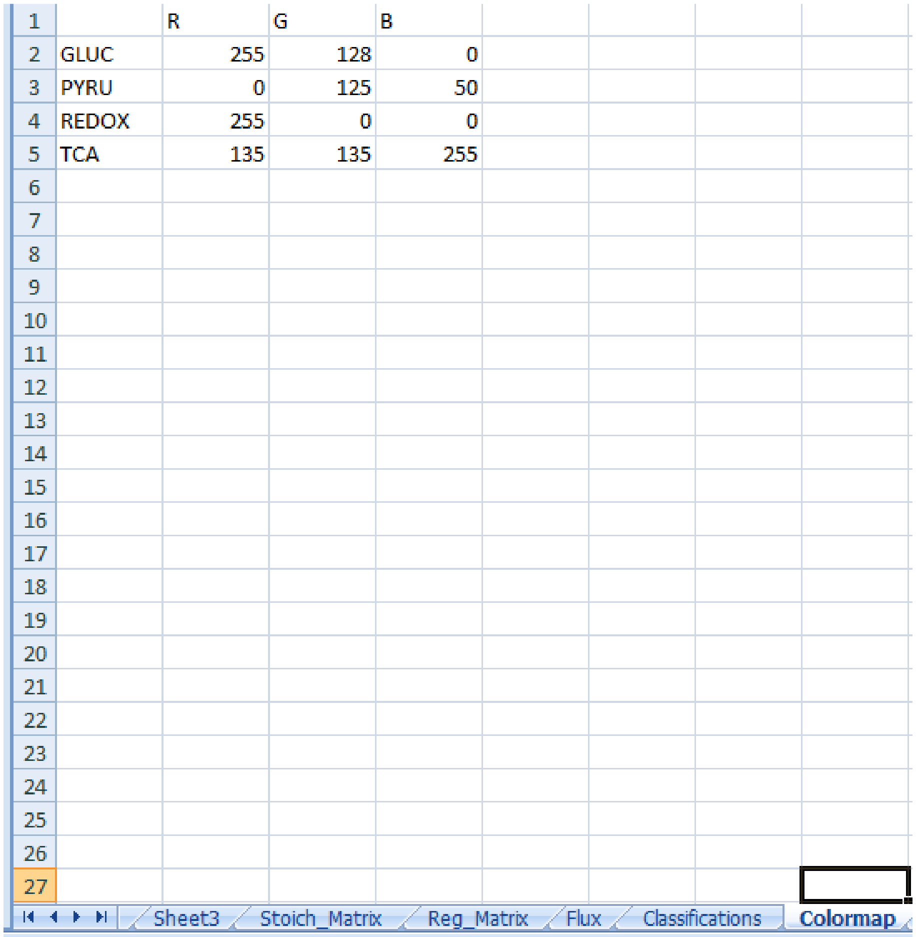
1	Reactions	Fluxes
2	R1	1
3	R2	1
4	R3	1
5	R4	1
6	R5	1
7	R6	1
8	R7	1
9	R8	1
10	R9	1
11	R10	1
12	R11	1
13	R12	1
14	R13	1
15	R14	1
16	R15	1
17	R16	1
18	R17	1
19	R18	1
20	R19	1
21	R20	1
22	R21	1
23	R22	1
24	R23	1
25	R24	1
26	R25	1
27	R26	1

Classifications: In this sheet, each reaction is classified to a canonical textbook metabolic pathway. This provides the information for the module pie charts, where each slice represents the fraction of reactions in the module that belong to a specific classified pathway. In this case, reactions were grouped as either glucose metabolism (GLUC), pyruvate metabolism (PYRU), redox reaction (REDOX), or TCA cycle.



1	R1	GLUC
2	R2	GLUC
3	R3	GLUC
4	R4	GLUC
5	R5	GLUC
6	R6	PYRU
7	R7	PYRU
8	R8	PYRU
9	R9	PYRU
10	R10	GLUC
11	R11	GLUC
12	R12	GLUC
13	R13	GLUC
14	R14	GLUC
15	R15	GLUC
16	R16	GLUC
17	R17	PYRU
18	R18	PYRU
19	R19	REDOX
20	R20	PYRU
21	R21	GLUC
22	R22	REDOX
23	R23	GLUC
24	R24	GLUC
25	R25	REDOX
26	R26	GLUC
27	R27	GLUC

Colormap: Here, each classification group is given a RGB color scale to denote what color the slice in the pie should be for the particular classification group.



	R	G	B
GLUC	255	128	0
PYRU	0	125	50
REDOX	255	0	0
TCA	135	135	255

Shred_Network()

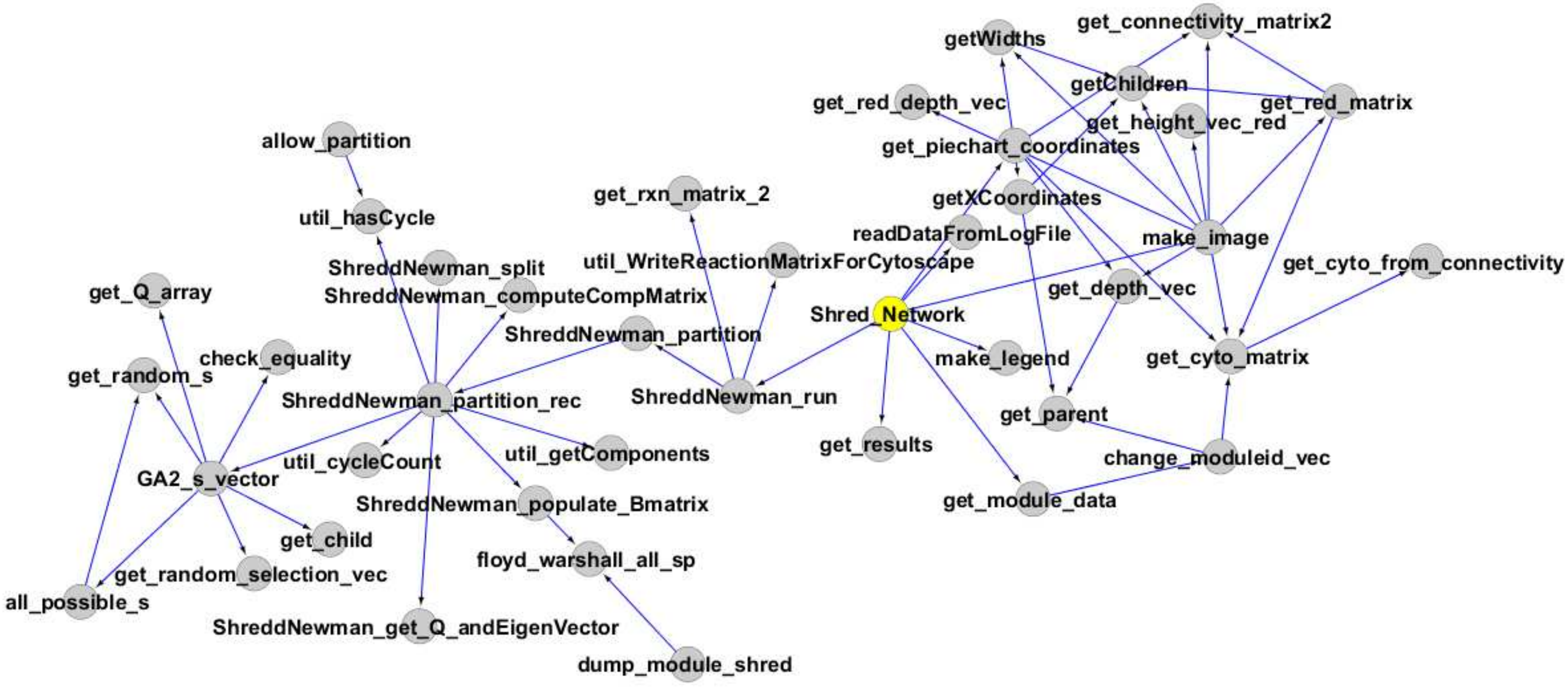
The Shred_Network function is the executable function for ShReD-based partitioning. The function definition is listed below.

```
function [raw_data,raw_result] =
Shred_Network(inputfilename,include_flux,include_visual)

%inputfilename  = The filename of the model. This is a string. For example,
%'my_model.xls'

%include_flux is a true/false boolean for whether or not flux data should
%be included

%include_visual is a true/false boolean for whether or not a visual should
%be created. Since the visual takes a long time to generate, it's not
%always desired.

%raw_data is a text file dump describing how each module gets partitioned.
%raw_result takes the raw data and provides a clean cell array table with
%each module ID and which reactions are present in each module.

global filename
filename = inputfilename;
global fid_counters;

%raw_rg are the reaction groupings.
[Output_1,Output_2,raw_rg]=xlsread(filename,'Classifications');
global network_size;
network_size = size(raw_rg,1);

fid_counters = fopen('counters.txt','w');

%Input the flux vector
[Output_1,Output_2,raw_fluxes]=xlsread(filename,'Flux');
fluxes = raw_fluxes(2:end,2);
fluxes = cell2mat(fluxes);

%Run the ShReD
ShreddNewman_run(filename,'Stoich_Matrix','Reg_Matrix','Shred_Output',false,f
alse,true,include_flux,fluxes)

%The ShReD produces all the raw data in a .log file, which then gets read
raw_data = readDataFromLogFile('Shred_Output.log');
xlswrite(filename,raw_data,'Raw_Data');

%The raw result takes the raw data and organizes a cell array with module
%ID and which reactions are present in that module.

raw_result = get_results(raw_data,raw_rg);
raw_result = change_raw_result(raw_result,raw_data);

[~,~,cmap]=xlsread(filename,'Colormap');
sheetinfo = cmap(2:end,1);
```

```
%sheetinfo = unique(raw_rg(:,2));
tally_vec = get_tally(filename);

%Making the visual
if(size(raw_data,1)>3)
    if(include_visual)
        [xcoords,ycoords,unique_ids]=get_piechart_coordinates(raw_data);

make_image(raw_data,raw_result,sheetinfo,tally_vec,xcoords,ycoords,filename);
        make_legend(filename,sheetinfo);
    end
    get_module_data(raw_data,raw_result,filename);
else
    fprintf('No Shred-Based Partitioning Possible\n');
end

delete('pie_temp.png');
delete('test_bar.png');
delete('test_text.png');
delete('test_matrix.png');
delete('text_matrix.png');
delete('bar_matrix.png')
```

ShReD_Network Function Calls

Function calls, originating from the executable Shred_Network(). The function definitions for each function in the graph is listed in alphabetical order. Function i points to function j if i calls j in the function definition.



allow_partition

In chapter 3, we only partitioned if both sub-graphs had at least one cycle

```
function bool = allow_partition(s,adjmatrix)

    vec_index_1 = find(s==1);
    vec_index_2 = find(s==-1);

    matrix1 = adjmatrix(vec_index_1,vec_index_1);
    matrix2 = adjmatrix(vec_index_2,vec_index_2);

     is_cycle_1 = 0;
        if(isempty(matrix1)~=1)
            is_cycle_1 = util_hasCycle(matrix1);
        end
        is_cycle_2 = 0;
        if(isempty(matrix2)~=1)
            is_cycle_2 = util_hasCycle(matrix2);
        end

    if(is_cycle_1 ==1 & is_cycle_2 ==1)
        bool = 1;
    else
        bool = 0;
    end
```

all_possible_s

```
function [max_Q,max_s] = all_possible_s(bmatrix)

%get max Q by testing all possible s vectors
% only for small adjacency matrices
global compare_GA_eig_fid;
%02/15/2012
max_Q = 0;
max_s = get_random_s(size(bmatrix,1));
for(i=0:2^size(bmatrix,1)-1)
    temp_s = zeros(size(bmatrix,1),1);
    binary_string = dec2bin(i);
    for(j=1:length(binary_string))
        if(str2num(binary_string(j))==1)
            temp_s(j,1) = 1;
        else
            temp_s(j,1) = -1;
        end
        Q_temp = temp_s'*bmatrix*temp_s;
        if(Q_temp > max_Q)
            max_Q = Q_temp;
            max_s = temp_s;
        end
    end
end
```

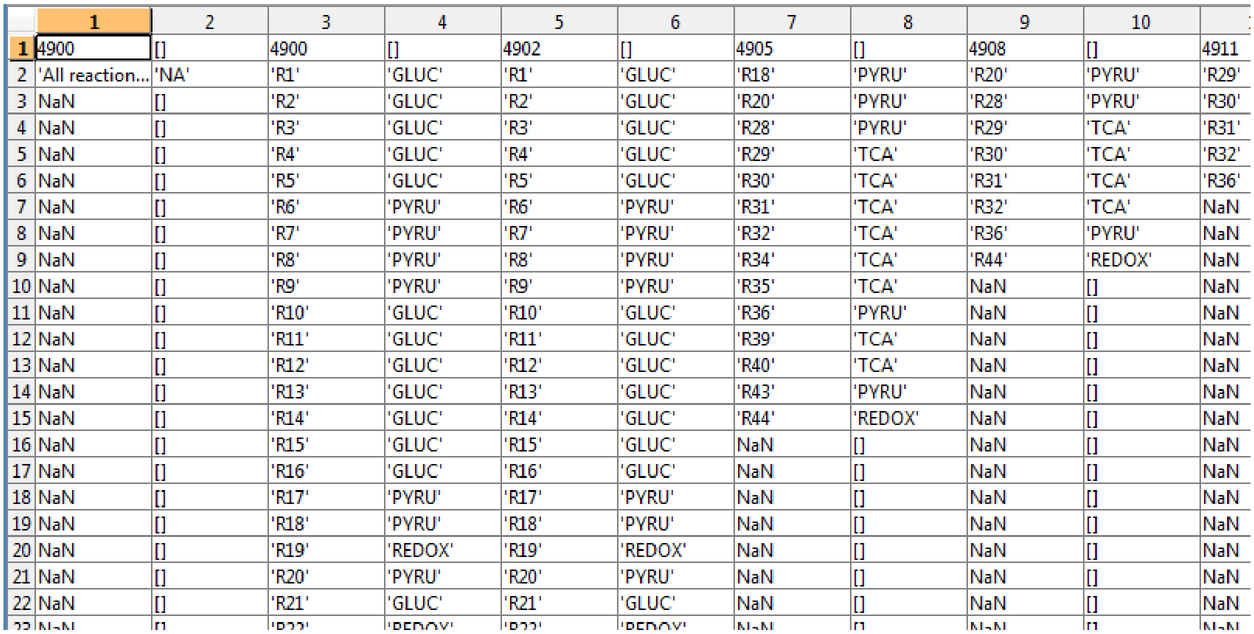
```
fprintf(compare_GA_eig_fid,'%i\t%f\t%f\n',size(bmatrix,1),max_Q,max_Q);
```

change_moduleid_vec

```
function new_vec = change_moduleid_vec(old_vec,raw)

% change the id of the modules to account for the artificial child node
creation in the
%hierarchy

new_vec = old_vec;
[cyto_new,cyto_old]=get_cyto_matrix(raw);
[matrix_old,unique_old] = get_connectivity_matrix(cyto_old);

for(i=1:length(old_vec))
   k = find(unique_old==old_vec(i));
   parent = get_parent(k,matrix_old);
   children_of_parent = getChildren(parent,matrix_old);
   if(length(children_of_parent)==1)
       new_vec(i) = unique_old(parent);
   end
end
```

change_raw_result

```
function raw_result_new = change_raw_result(raw_result,raw_data)

%Change module IDs to account for artificial child node creation in the
%hierarchy.

vec = raw_result(1,:);
index = 1;
for(i=1:2:length(vec))
    vec_new(index) = cell2mat(vec(i));
    index = index + 1;
end

vec_new_changed = change_moduleid_vec(vec_new,raw_data);

index = 1;
for(i=1:2:length(vec))
    vec_new_changed_cell(i) = num2cell(vec_new_changed(index));
    index = index + 1;
end
vec_new_changed_cell(i+1)=cellstr('');
raw_result(1,:) = vec_new_changed_cell;
raw_result_new = raw_result;
```



	1	2	3	4	5	6	7	8	9	10	
1	4900	[]	4900	[]	4902	[]	4905	[]	4908	[]	4911
2	'All reaction...	'NA'	'R1'	'GLUC'	'R1'	'GLUC'	'R18'	'PYRU'	'R20'	'PYRU'	'R29'
3	NaN	[]	'R2'	'GLUC'	'R2'	'GLUC'	'R20'	'PYRU'	'R28'	'PYRU'	'R30'
4	NaN	[]	'R3'	'GLUC'	'R3'	'GLUC'	'R28'	'PYRU'	'R29'	'TCA'	'R31'
5	NaN	[]	'R4'	'GLUC'	'R4'	'GLUC'	'R29'	'TCA'	'R30'	'TCA'	'R32'
6	NaN	[]	'R5'	'GLUC'	'R5'	'GLUC'	'R30'	'TCA'	'R31'	'TCA'	'R36'
7	NaN	[]	'R6'	'PYRU'	'R6'	'PYRU'	'R31'	'TCA'	'R32'	'TCA'	NaN
8	NaN	[]	'R7'	'PYRU'	'R7'	'PYRU'	'R32'	'TCA'	'R36'	'PYRU'	NaN
9	NaN	[]	'R8'	'PYRU'	'R8'	'PYRU'	'R34'	'TCA'	'R44'	'REDOX'	NaN
10	NaN	[]	'R9'	'PYRU'	'R9'	'PYRU'	'R35'	'TCA'	NaN	[]	NaN
11	NaN	[]	'R10'	'GLUC'	'R10'	'GLUC'	'R36'	'PYRU'	NaN	[]	NaN
12	NaN	[]	'R11'	'GLUC'	'R11'	'GLUC'	'R39'	'TCA'	NaN	[]	NaN
13	NaN	[]	'R12'	'GLUC'	'R12'	'GLUC'	'R40'	'TCA'	NaN	[]	NaN
14	NaN	[]	'R13'	'GLUC'	'R13'	'GLUC'	'R43'	'PYRU'	NaN	[]	NaN
15	NaN	[]	'R14'	'GLUC'	'R14'	'GLUC'	'R44'	'REDOX'	NaN	[]	NaN
16	NaN	[]	'R15'	'GLUC'	'R15'	'GLUC'	NaN	[]	NaN	[]	NaN
17	NaN	[]	'R16'	'GLUC'	'R16'	'GLUC'	NaN	[]	NaN	[]	NaN
18	NaN	[]	'R17'	'PYRU'	'R17'	'PYRU'	NaN	[]	NaN	[]	NaN
19	NaN	[]	'R18'	'PYRU'	'R18'	'PYRU'	NaN	[]	NaN	[]	NaN
20	NaN	[]	'R19'	'REDOX'	'R19'	'REDOX'	NaN	[]	NaN	[]	NaN
21	NaN	[]	'R20'	'PYRU'	'R20'	'PYRU'	NaN	[]	NaN	[]	NaN
22	NaN	[]	'R21'	'GLUC'	'R21'	'GLUC'	NaN	[]	NaN	[]	NaN
23	NaN	[]	'R22'	'REDOX'	'R22'	'REDOX'	NaN	[]	NaN	[]	NaN

The output here is the raw_result, which has each module ID in the top row with which reactions are contained in the module below along with its canonical textbook function in the adjacent column.

check_equality

```
function is_equal = check_equality(s_vec,pop_matrix);

%determine if an s vector or its negation is present in the population of
%svectors
is_equal = 0;
for(i=1:size(pop_matrix,2))
    if(isequal(s_vec,pop_matrix(:,i)))
        is_equal = 1;
    end
    if(isequal(-s_vec,pop_matrix(:,i)))
        is_equal = 1;
    end
end
```

dump_module_shred

Input: Module adjacency matrix

Output: Textfile output for all ShReD paths with cofactor and non-cofactor interactions.

Floyd_warshall_all_sp

Input: Sparse directed graph A.
Output: Matrix of all pairs of shortest paths from node i to node j.

GA2_s_vector

```
function [total_max_Q,s_return,generation] = GA_s_vector(bmatrix,adjmatrix)

global compare_GA_eig_fid;
```

```
%bmatrix = either G or V matrix from ShReD
%adjmatrix = adjacency matrix
%total_max_Q = the maximum Q score of all s vectors in the population over
%the GA
%s_return = the s_vector that gave the maximum Q
% genration = the number of generations before the GA terminates.

%This version doesn't make sure the partition yields modules with two
%cycles
% parameters for the GA:

%population size is the number of s_vectors in the population.
if(size(bmatrix,1)<9)
    [total_max_Q,s_return]=all_possible_s(bmatrix);
else
    population_size = 100;

    % selection is the fraction of the population that gets killed each
    % generation due to lack of fitness. Basically the fraction of s vectors
    % that give the least Q score are removed.
    selection = 0.4;

    %mutate percentage is the fraction of the population after selection that
    %is subject to mutation.
    mutate_percentage = 0.2;

    %mutate vars is the number of variables that get mutated for the
s_vectors
    %that were selected to get mutated.
    mutate_vars_orig = 2;

    keep_ratio = 0.1;

    global fid_counters;

    %initialize the population as mXn matrix where m is the number of
elements
    %in the s_vector and n is the number of s_vectors. Each column is an
    %individual in the population. The s_vectors are random, but are allowed
    %only if a partition is possible as a result.
    len = size(bmatrix,1);
    pop_matrix = zeros(len,population_size);

    total_max_Q = 0;
    s_return = get_random_s(len);

    pop_possible = 1;
    for(i=1:population_size)
        is_allow = 0;
        count_iterations = 0;
        while(is_allow==0)
            s_rand = get_random_s(len);
            if(check_equality(s_rand,pop_matrix)==0)
                is_allow = 1;
            end
            count_iterations = count_iterations + 1;
            if(count_iterations>size(bmatrix,1)*100)
                is_allow = 1;
                pop_possible = 0;
            end
        end
        pop_matrix(:,i) = s_rand;
    end

    if(pop_possible == 1)
        is_loop = 1;
        generation = 1;
        while(is_loop == 1)
            % Obtain Q score for all the s_vectors in the population, sort
them,
            % and figure out which s_vectors need to be killed off based on
the
            % selection percentage.
            Q_array = get_Q_array(pop_matrix,bmatrix);
            [Q_array_sorted,Q_sort_index]=sort(Q_array);
            num_remove = ceil(population_size*selection);
            num_keep = ceil(keep_ratio*num_remove);

            % Keep random keep_percent of the bad population.
            remove_index = Q_sort_index(1:num_remove);
            keep_vec =
get_random_selection_vec(length(remove_index),num_keep);
            remove_index(keep_vec) = [];

            % Remove the unfit solutions
            pop_matrix(:,remove_index) = [];

            %Determine which s_vectors should be subject to a mutation
            num_mutate = ceil(size(pop_matrix,2)*mutate_percentage);
            pick_mutate_s_vec =
get_random_selection_vec(size(pop_matrix,2),num_mutate);


            %mutate each s_vec
            for(i=1:length(pick_mutate_s_vec))
                is_allow = 0;
                counter = 0;
                while(is_allow==0)
                    %The number of variables to mutate can be one more or one
                    %less the specified value
                    x = ceil(rand()*3);
                    if(x==1)
                        mutate_vars = mutate_vars_orig - 1;
                    elseif(x==2)
                        mutate_vars = mutate_vars_orig;
                    elseif(x==3)
                        mutate_vars = mutate_vars_orig + 1;
                    end
                    pick_mutate_vars_vec =
get_random_selection_vec(size(pop_matrix,1),mutate_vars);
                    for(j=1:length(pick_mutate_vars_vec))
```

```
                        s_mutated = pop_matrix(:,pick_mutate_s_vec(i));
                        if(s_mutated(pick_mutate_vars_vec(j)) == 1)
                            s_mutated(pick_mutate_vars_vec(j)) = -1;
                        elseif(s_mutated(pick_mutate_vars_vec(j)) == -1)
                            s_mutated(pick_mutate_vars_vec(j)) = 1;
                        end
                    end
                    %ensure that the mutated s_vectors still allow for a
partition
                    if(check_equality(s_mutated,pop_matrix)==0)
                        is_allow = 1;
                    end
                     counter = counter +1;
                     if(counter>10*size(bmatrix,1))
                         is_allow = 1;
                         boom = 1;
                          fprintf(fid_counters,'Bmatrix:%i\t Genration:%i\t
Size of Pick Mutate Vec:%i\t Iteration:%i
\n',size(bmatrix,1),generation,length(pick_mutate_s_vec),i);
                     end
                end
                if(counter<=100*size(bmatrix,1))
                    pop_matrix(:,pick_mutate_s_vec(i)) = s_mutated;
                end

            end

            %Get next generation population. For a population size number of
            %iterations, randomly select two parents from the original
population
            %and have them mate. The get_child function takes two parents,
and
            %produces a child - if the element of both parents is the same,
the
            %child retains that element, but if they are different, then
which
            %parent's trait to retain is randomly chosen.

            pop_matrix_new = zeros(size(bmatrix,1),1);
            for(i=1:population_size)
                is_allow = 0;
                counter = 0;
                while(is_allow == 0)
                    parents = get_random_selection_vec(size(pop_matrix,2),2);
                    parent_1 = pop_matrix(:,parents(1));
                    parent_2 = pop_matrix(:,parents(2));
                    s_child = get_child(parent_1,parent_2);
                    if(check_equality(s_child,pop_matrix_new)==0)
                        is_allow = 1;
                    end
                    counter = counter + 1;

                end
                pop_matrix_new(:,i) = s_child;

            end
            pop_matrix = pop_matrix_new;
            Q_matrix(generation,:) = Q_array;
            Q_ave(generation) = mean(Q_array);
            generation = generation + 1;
            fit = 1;
            if(generation>101)
                a = [1:length(Q_ave)];
                b = Q_ave;

                c = a(length(a)-100:length(a));
                d = b(length(b)-100:length(b));
                fit = polyfit(c,d,1);
            end

            Q_array_new = get_Q_array(pop_matrix,bmatrix);
            max_Q = max(Q_array_new);
            max_Q_index = find(Q_array_new == max_Q);
            s_GA_optimal = pop_matrix(:,max_Q_index(1));
            if(max_Q>total_max_Q)
                total_max_Q = max_Q;
                s_return = s_GA_optimal;
            end
            generation
            fit(1)


            %Terminate the GA after some number of generations. This
termination
            %criterion should change based on some function of the elbow arm.
            if(abs(fit(1))<0.05)
                is_loop = 0;
                generation
            end
        end

        % Keep track of the average and stdev Q at each generation so that Q
can be
        % plotted as a function of a generation.
        for(i=1:size(Q_matrix,1))
            Q_matrix_plot(i,1) = i;
            Q_matrix_plot(i,2) = mean(Q_matrix(i,:));
            Q_matrix_plot(i,3) = std(Q_matrix(i,:));
        end
        s_eig = get_s_eig(bmatrix);
        Q_eig = s_eig'*bmatrix*s_eig;
        Q_matrix_plot;

fprintf(compare_GA_eig_fid,'%i\t%f\t%f\n',size(bmatrix,1),total_max_Q,Q_eig);
    else
        max_Q = 0;
        s_GA_optimal = get_random_s(len);
        fprintf(compare_GA_eig_fid,'%i\t%f\t%f\n',size(bmatrix,1),0,0);
    end
```

```
end
```

generate_s_vec

generate a random s vector, used for the genetic algorithm

```
function s = generate_s_vec(length);

for(i=1:length)
    r = rand();
    if(r>0.50)
        s(i) = 1;
    else
        s(i) = -1;
    end
end
```

get_connectivity_matrix2

Input: two column matrix (cyto_matrix) with parent module in the left column and child module in the right column.

Output: connectivity matrix of the module hierarchy based on the cyto_matrix

```
function [matrix,unique_ids] = get_connectivity_matrix2(cyto_matrix)

if(iscell(cyto_matrix(1,1))~=1)
    cyto_matrix = num2cell(cyto_matrix);
end
        col_1 = cyto_matrix(:,1);
        col_2 = cyto_matrix(:,2);
        col_total = [col_1;col_2];


    for(i=1:length(col_total))
         col_total_real(i)=cell2mat(col_total(i));
    end

% col_total_real = [root;col_total_real(find(col_total_real)~=root)'];
unique_ids = unique(col_total_real);
length_matrix = length(unique_ids);

%Create Matrix - length of the unique_id matrix

matrix = zeros(length_matrix,length_matrix);

%For each row in cyto_matrix, place a 1 for the connectivity in the matrix
%by comparing the strings (get_matrix_index)
```

```
for(i=1:size(cyto_matrix,1))

    id_1 = cell2mat(cyto_matrix(i,1)); % parent
    id_2 = cell2mat(cyto_matrix(i,2)); %child

    index_1 = find(unique_ids==id_1); %index of unique ID where parent is
    index_2 = find(unique_ids==id_2); %index of uniqueIDs where child is

    matrix(index_1,index_2) = 1; %matrix(parent,child)
end
```

get_child

```
function s_child = get_child(parent_1,parent_2)

%part of genetic algorithm (GA2_s_vector)
%get child of two parent s vectors
for(i=1:length(parent_1))
    if(parent_1(i) == parent_2(i))
        s_child(i) = parent_1(i);
    else
        x = rand();
        if(x>0.5)
            s_child(i) = parent_1(i);
        else
            s_child(i) = parent_2(i);
        end
    end
end
```

getChildren

Input: Module ID and connectivity matrix

Output: children of the module

```
function children = getChildren(index,matrix)
children = find(matrix(index,:));
```

get_cyto_matrix

cyto_matrix is basically a two column matrix with left column for parent modules and right column for child modules. The old cyto matrix is based on the original partition data, and the new cyto matrix is after removing artificial nodes as explained in get_results.

```
function [cyto_matrix_new,cyto_matrix_old] = get_cyto_matrix(raw)

%Get original cytomatrix = cytomatrix_old
matrix_index = 1;
size_r = size(raw);
for(i=2:size_r(1))
```

```
    cyto_matrix(matrix_index,1) = raw(i,4);
    cyto_matrix(matrix_index,2) = raw(i,1);
    matrix_index = matrix_index +1;

    char_happens = char(raw(i,6));
    if(char_happens(1)=='d')
        c = textscan(char_happens,'%s %s %d %s %d');
        cyto_matrix(matrix_index,1)=raw(i,1);
        cyto_matrix(matrix_index,2)=c(3);
        matrix_index=matrix_index+1;
        cyto_matrix(matrix_index,1)=raw(i,1);
        cyto_matrix(matrix_index,2)=c(5);
        matrix_index = matrix_index + 1;
    end
end
cyto_matrix_old = cyto_matrix;

%Fix the cytomatrix to get cyto_matrix_new
[matrix_old,unique_ids]=get_connectivity_matrix(cyto_matrix);
matrix_new = remove_redundancies(matrix_old);
sum_entries = 0;
for(i=1:size(matrix_old,1))
    for(j=1:size(matrix_old,2))
        if(matrix_old(i,j)==1)
            sum_entries = sum_entries + 1;
        end
    end
end



if(sum_entries==1)
    cyto_matrix_old(1,1)=num2cell(0);
    for(i=1:size(cyto_matrix_old,1))
        for(j=1:size(cyto_matrix_old,2))
            cyto_matrix_new(i,j) = cyto_matrix_old(i,j);
        end
    end
else
   cyto_matrix_new = get_cyto_from_connectivity(matrix_new,unique_ids);
end

%Remove situation where node's parent is itself.
rem_vec(1) = 100;
if(sum_entries~=1)
    rem_index = 1;
    for(k=1:size(cyto_matrix_new,1))
        if(cyto_matrix_new(k,1)==cyto_matrix_new(k,2))
            rem_vec(rem_index)=k;
            rem_index = rem_index + 1;
        end
    end
    if(rem_vec(1)~=100)
        cyto_matrix_new(rem_vec,:)=[];
        cyto_matrix_new = num2cell(cyto_matrix_new);
    end
end
```

get_cyto_from_connectivity

```
function cyto_new = get_cyto_from_connectivity(matrix,unique_ids)

index = 1;
for(i=1:size(matrix,1))
    for(j=1:size(matrix,2))
        if(matrix(i,j)==1)
            cyto_new(index,1)=unique_ids(i); %parent
            cyto_new(index,2)=unique_ids(j); %child
            index = index + 1;
        end
    end
end
```

get_depth_vec

```
function depth_vec  = get_depth_vec(matrix)

size_matrix = size(matrix);
depth_vec = ones(size_matrix(1),1);

for(i=2:size_matrix(1))
    parent_index = i;
    while(parent_index~=1)
        parent_index = get_parent(parent_index,matrix);
        depth_vec(i) = depth_vec(i) + 1;
    end
end

depth_vec = depth_vec -1;
```

get_height_vec_red

```
function [heights] = get_height_vec_red(m , red_matrix)

%get height of module but only ShReD-based partitions count
%modules breaking into connected components are denoted by red lines
n = size(m,1);
heights = zeros(n,1);

parents = zeros(n,1);
for i = 1:n
  for j = 1:n
    if (m(i,j) == 1)
      parents(j) = i;
    end
  end
end
```

```
rootNode = find(parents == 0);

heights = get_red_heights(m,red_matrix,parents,rootNode);

end

function [heights] = get_red_heights(m, red_matrix, parents , i)

  n = size(m,1);
  heights = zeros(n,1);
  children = getChildren(i,m);
  maxHeight = 0;

  for (j = 1:length(children))
    child = children(j);
    newHeights = get_red_heights(m,red_matrix,parents,child);
    heights = heights + newHeights;
    heightOption = newHeights(child);
    if (red_matrix(i,child) < 1)
      heightOption = heightOption + 1;
    end
    if (heightOption > maxHeight)
      maxHeight = heightOption;
    end
  end
  heights(i) = maxHeight;
end
```

get_module_data

```
function A = get_module_data(raw,raw_result,filename)

%A outputs a cell array of module data, including id, size, cycle count,
%etc.
fid = fopen('ave_std_shredd.txt');
lines = textscan(fid,'%s','delimiter','\n');
lines = lines{1,1};
for(i=2:size(lines,1))
    line = lines(i);
    line = line{1};
    line = str2num(line);
    B(i-1,:) = line;
end

old_vec = B(:,1);
new_vec = change_moduleid_vec(old_vec,raw);
B(:,1) = new_vec;
A =xlsread('Shred_Example_Data.xls');

% A is the larger matrix
% B has the shred information

for(i=1:size(B,1))
    module_id = B(i,1);
    k=find(A(:,1)==module_id);
    A(k,10)=B(i,2);
    A(k,11)=B(i,3);
    A(k,12)=B(i,4);
end

A(:,6) = [];
header(1,1) = cellstr('Module_ID');
header(1,2) = cellstr('Size');
header(1,3) = cellstr('Cycle_Count');
header(1,4) = cellstr('Cycle/Size');
header(1,5) = cellstr('Number of Canonical Groupings');
header(1,6) = cellstr('Homogeneity Index');
header(1,7) = cellstr('Depth');
header(1,8) = cellstr('Height');
header(1,9) = cellstr('Average_Shred');
header(1,10) = cellstr('Standard_Deviation_Shred');
header(1,11) = cellstr('Number of Shreds');

xlswrite(filename,header,'Module_Report','A1')
xlswrite(filename,A,'Module_Report','A2');
delete('Shred_Example_Data.xls');

vec = raw_result(1,:);
index = 1;
for(i=1:2:length(vec))
    vec_new(index) = cell2mat(vec(i));
    index = index + 1;
end

vec_new_changed = change_moduleid_vec(vec_new,raw);

index = 1;
for(i=1:2:length(vec))
    vec_new_changed_cell(i) = num2cell(vec_new_changed(index));
    index = index + 1;
end
vec_new_changed_cell(i+1)=cellstr('');
raw_result(1,:) = vec_new_changed_cell;

xlswrite(filename,raw_result,'Module_Composition');
[~,~,modules]=xlsread(filename,'Module_Composition');
```

get_module_reactions

Input: module id, raw_result

output: vector of reaction ids contained in the module

get_parent

```
function parent_index = get_parent(source_index,matrix)

%Get parent of a module (source_index) in the hierarchy
column = matrix(:,source_index);
parent_index = find(column);
```

get_piechart_coordinates

For each module, obtain the x and y coordinate for where the pie chart should be placed in the image.

```
function [xCoords,yCoords,unique_ids]=get_piechart_coordinates(raw)

y_scale = 180;
x_scale = 240;

[cyto_matrix_new,cyto_matrix_old] = get_cyto_matrix(raw);
[matrix,unique_ids] = get_connectivity_matrix2(cyto_matrix_new);
depth_vec = get_depth_vec(matrix);
width_vec = getWidths(depth_vec,matrix,x_scale);
xCoords = getXCoordinates(width_vec,depth_vec,matrix)+0.5*width_vec;
yCoords = y_scale*depth_vec+100;
```

get_Q_array

```
function Q_array = get_Q_array(pop_matrix,bmatrix)

for(i=1:size(pop_matrix,2))
    s_i = pop_matrix(:,i);
    Q_array(i) = s_i'*bmatrix*s_i;
end
```

get_random_s

```
function s_rand = get_random_s(len)

for(i=1:len)
    a = rand();
    if(a>0.50)
        s_rand(i) = 1;
    else
        s_rand(i) = -1;
    end
end

s_rand = s_rand';
```

get_random_selection_vec

```
function selection_vec = get_random_selection_vec(n,k)
% Part of genetic algorithm GA2_s_vector
vec = [1:n];
for(i=1:k)

    x =ceil(rand()*length(vec));
    selection_vec(i) = vec(x);
    index = find(vec == selection_vec(i));
    vec(index) = [];
end
```

get_red_depth_vec

```
function [depths] = get_red_depth_vec(m , red_matrix)
%Get depth of module but only ShReD-based partitions count, not breaking
%into components (as denoted by red lines in the hierarchy)

n = size(m,1);
depths = zeros(n,1);


parents = zeros(n,1);
for i = 1:n
  for j = 1:n
    if (m(i,j) == 1)
      parents(j) = i;
    end
  end
end
rootNode = find(parents == 0);

depths = get_red_depths(m,red_matrix,parents,rootNode,0);

end

function [depths] = get_red_depths(m, red_matrix, parents , i,d)
  n = size(m,1);
  depths = zeros(n,1);
  depths(i) = d;
  children = getChildren(i,m);

  for (j = 1:length(children))
    child = children(j);
    newD = d;

    if (red_matrix(i,child) < 1)
      newD = newD + 1;
    end
    depths = depths + get_red_depths(m,red_matrix,parents,child,newD);
  end
end
```

get_red_matrix

```
function [red_matrix,matrix_new,unique_new] = get_red_matrix(raw);

%Determine if an edge between two modules in the hierarchy is based on
%breaking down into connected components (not Shred-based partitioning).
%Information gets stored in a matrix.
[cyto_matrix_new,cyto_matrix_old]=get_cyto_matrix(raw);
[matrix_old,unique_old]=get_connectivity_matrix(cyto_matrix_old);
[matrix_new,unique_new]=get_connectivity_matrix2(cyto_matrix_new);

red_matrix = zeros(length(unique_new),length(unique_new));

raw_mat = raw(3:size(raw,1),1:4);
raw_mat = cell2mat(raw_mat);

vec = raw_mat(:,1);
for(i=1:length(unique_new))
    k = find(unique_new(i)==vec);
    if(raw_mat(k,1)==raw_mat(k,4))
        children = getChildren(i,matrix_new);
        for(j=1:length(children))
            red_matrix(i,children(j))=1;
        end
    end
end
```

getXCoordinates

```
function xcoords = getXCoordinates(width_vec,depth_vec,matrix)

maxDepth = max(depth_vec);
numNodes = size(matrix,1);
xcoords = zeros(numNodes,1);
for depth = 0:maxDepth
    for i = 1:numNodes
        if (depth_vec(i) == depth)
            if (depth == 0)
                xcoords(i) = 0;
            else
                parent = get_parent(i,matrix);
                xcoordSum = xcoords(parent);
                for child = getChildren(parent,matrix)
                    if (child < i)
                        xcoordSum = xcoordSum + width_vec(child);
                    end
                end
                xcoords(i) = xcoordSum;
            end
        end
    end
end
```

get_results

```
function concat_2 = get_results(raw,rg)
%raw = raw_data file
%rg -first column = reaction, 2nd column = reaction type.
% The output is module ID on top and all the reactions contained in that
% module below along with the reaction grouping to the side.

size_r = size(raw);

id = raw(2:size_r(1),1);
modules = raw(2:size_r(1),7:size_r(2));
id = id';
modules = modules';
concat = [id;modules];
%concat has the IDs in the first row and the reactions for each module in
%columns.

size_c = size(concat);
size_rg = size(rg);
vec(1)=cellstr('NA');
for(i=1:size_c(2))
    concat_2(:,2*i-1) = concat(:,i);
    isloop = 1;
    index = 1;
    while(isloop==1)
        a = concat(index+1,i);
        isloop_2 = 1;
        index_2 = 1;
        while(isloop_2==1)
            if(strcmp(rg(index_2,1),a)==1)
                %vec stores what type the given reaction is
                vec(index) = rg(index_2,2);
                isloop_2 =0;
            end
            index_2 = index_2 + 1;
            if(index_2>size_rg(1))
                isloop_2=0;
            end
        end
        if(isnan(cell2mat(a))==1)
            isloop = 0;
        end
        index = index + 1;
        if(index+1>size_c(1))
            isloop=0;
        end
        %Have found all reaction types.
    end
    size_vec = size(vec);
    if(size_vec(1)==1)
        vec = vec';
    end
    size_vec = size(vec);
    concat_2(2:size_vec(1)+1,2*i) = vec;
    clear vec
    vec(1)=cellstr('none');
end
```

get_rxn_matrix_2

```
function [ ReactionAdjMatrix, interaction_matrix, ReactionLabels,
numReactions ] =get_rxn_matrix_2( inputFile, inputSheet, fluxes,
regulationSheet, regulation)

% Obtain reaction adjacency matrix with accounting for flux weights.
% Called by ShreddNewman_run if flux weights are used.
global filename;
r =regexp (inputFile, '\.', 'split');
if (regulation)
    s= strcat(r(1), '_reactionRegCyto.txt' );
else
    s= strcat(r(1), '_reactionCyto.txt' );
end
fid = fopen(char(s(1)), 'w+');
fid_cyto = fopen('rxn_cyto.txt','w+');
% read from xls
[numData, textData] = xlsread(inputFile, inputSheet);




reversibility = numData(1,:);
adjMatrix = numData(3:end, :);
numMetabolites = size (adjMatrix,1);
numReactions = size(adjMatrix, 2);
ReactionLabels(1:numReactions) = textData(2,3:end);

formattedTextData(1:numMetabolites) = textData(3:end,2);
formattedTextData(numMetabolites + 1:numMetabolites + numReactions) =
textData(2,3:end);

%08/24/11 - changed inf*ones to zeros
ReactionAdjMatrix = inf*ones(numReactions,numReactions);
interaction_matrix = zeros(numReactions,numReactions);






fid_distance = fopen('flux_distances.txt','w+');
for i = 1: numReactions
    column =  adjMatrix (:, i);
    assert (sum (column < 0) > 0); % at least one negative entry in the
column
```

```
    potential_metabolites = find ((column ~= 0));

    for j = 1: size (potential_metabolites, 1)
        if((fluxes(i)>0 & column(potential_metabolites(j))>0) || (fluxes(i)<0
& column(potential_metabolites(j))<0))
            row = adjMatrix(potential_metabolites(j),:);
            for(k=1:length(row))
                if((fluxes(k)>0 & row(k)<0)||(fluxes(k)<0 & row(k)>0))
                    interaction_matrix(i,k) = 1;
                    fprintf(fid_cyto,'R%i\tR%i\n',i,k);
                    fprintf(fid,'First Reaction:%i\tSecond
Reaction:%i\tMetabolite:%i\t',i,k,potential_metabolites(j));
                    total_consumed = 0;
                    for(l=1:length(row))
                        if((fluxes(l)>0 & row(l)<0)||(fluxes(l)<0 &
row(l)>0))
                            total_consumed = total_consumed -
row(l)*fluxes(l);
                            fprintf(fid,'Flux(%i):%f\t',l,-row(l)*fluxes(l));
                        end
                    end
                    distance = total_consumed/-((fluxes(k)*row(k)));
                    fprintf(fid,'Distance:%f\n',distance);
                    fprintf(fid_distance,'%f\n',distance);
                    if(ReactionAdjMatrix(i,k)>distance)
                        ReactionAdjMatrix(i,k) = distance;
                        intermediate(i,k) = potential_metabolites(j);
                    end
                end
            end
        end
    end
end

fclose(fid_distance);
%Regulation
[reg_num,~,reg]=xlsread(inputFile,regulationSheet);
for(i=1: numReactions)
    reg_column = reg_num(:,i);
    regulators = find(reg_column~=0);
    for(j=1:length(regulators))
        row = adjMatrix(regulators(j),:);
        for(k=1:length(row))
            if((row(k)>0 & fluxes(k)>0)||(row(k)<0 & fluxes(k)<0))
                ReactionAdjMatrix(k,i) = 1;
                interaction_matrix(k,i) = 1;
                fprintf(fid,'R%i\tREG\tR%i\tvia metabolite
%s\n',k,i,char(reg(regulators(j)+1,1)));
                fprintf(fid_cyto,'R%i\tR%i\n',k,i);
            end
        end
    end
end
```

```
ReactionAdjMatrix = ReactionAdjMatrix  - diag (diag(ReactionAdjMatrix));

% ADded 08/25/2011
for(i=1:size(ReactionAdjMatrix,1))
    for(j=1:size(ReactionAdjMatrix,2))
        if(ReactionAdjMatrix(i,j)==inf || isnan(ReactionAdjMatrix(i,j))==1)
            ReactionAdjMatrix(i,j) = 0;
        end
    end
end
xlswrite(filename,intermediate,'intermediate');
fclose(fid);
```

getWidths

Calculate the width each pie chart will take in the hierarchical tree.

```
function width_vec = getWidths(depth_vec,matrix,x_scale)

maxDepth = max(depth_vec);
numNodes = size(matrix,1);
width_vec = zeros(numNodes,1);
for depth = maxDepth:-1:0
    for i = 1:numNodes
        if (depth_vec(i) == depth)
            widthSum = 0;
            children = getChildren(i,matrix);
            if (size(children,2) == 0)
                width_vec(i) = x_scale;
            else
                for child = getChildren(i,matrix)
                    widthSum = widthSum + width_vec(child);
                end
                width_vec(i) = widthSum;
            end
        end
    end
end
```

make_image

Make the image for the hierarchical tree of modules (pie charts)

```
function make_image
(raw,raw_result,sheetinfo,tally_vec,xCoords,yCoords,filename)

y_scale = 180;
x_scale = 240;
[cyto_matrix_new,cyto_matrix_old] = get_cyto_matrix(raw);
[matrix,unique_ids] = get_connectivity_matrix2(cyto_matrix_new);
[matrix_old,unique_ids_old]=get_connectivity_matrix(cyto_matrix_old);
```

```
depth_vec = get_depth_vec(matrix);
num_nodes = size(depth_vec,1);
width_vec = getWidths(depth_vec,matrix,x_scale);


A = im2uint8(255*ones((2+max(depth_vec))*y_scale,width_vec(1),3));

[raw_new,visual_data]=get_quant_results(raw_result,sheetinfo,tally_vec);

%Remove the first line of visual_data so that the node doesn't get a
%uniform composition:

visual_data(1,:) = [];
pie([1,1])
visual_data = cell2mat(visual_data);
data_matrix = raw(2:size(raw,1),1:3);
data_matrix = cell2mat(data_matrix);
% Get rid of top redundancy
data_matrix(1,:) = [];

%datamatrix holds id, size, cyclecount
for i = 1:num_nodes
   composition_index = find(visual_data(:,1) == unique_ids(i));
   k = find(data_matrix(:,1)==unique_ids(i));
   if(size(composition_index,1)==0)
       new_id = unique_ids(i);
       %If it has no composition get its child's composition (in original
       %heirarchy, before collapsing it.
       old_id_index = find(unique_ids_old==new_id);
       child = getChildren(old_id_index,matrix_old);
       if(size(child,2)==1)
           composition_index = find(visual_data(:,1)==unique_ids_old(child));
           k = find(data_matrix(:,1)==unique_ids_old(child));
       end
   end

   if(size(composition_index,1)==0)
       composition = zeros(length(sheetinfo),1);
       composition(1) = 1;
   else
       composition = visual_data(composition_index,2:(size(visual_data,2)));
   end

   module_id = unique_ids(i);
   size_module = data_matrix(k,2);
   cycle_count = data_matrix(k,3);

    module_id = unique_ids(i);
   size_module = data_matrix(k,2);
   cycle_count = data_matrix(k,3);
   report(i,1)=module_id;
   report(i,2) = size_module(1);
   report(i,3) = cycle_count(1);
   report(i,4) = cycle_count(1)/size_module(1);
   report(i,5) = length(sheetinfo)-sum(composition==0);
   report(i,6) = max(composition)>50; %homogeneity index
   report(i,7) = max(composition)/100; %dominance factor

%    id_string = strcat('Size:',num2str(size_module),'
Cycles:',num2str(cycle_count));
   id_string = strcat('ID:',num2str(module_id),' Size:',num2str(size_module),
' Cycles:',num2str(cycle_count(1)));

map = xlsread(filename,'Colormap');
map = map/255;

%labeling
for(p=1:size(map,1))
    labeling(p) = cellstr('');
end
   pie(composition+0.00001,labeling);
   colormap(map);
   handle = gcf;

   find_vec = find(composition~=0);
   if(size(find_vec,2)==1)
       annotation(handle,'line',[0.517073170731707 0.517073170731707],...
        [0.513376996805112
0.853035143769968],'Color',map(find_vec,:),'LineWidth',8);
   end
   annotation(handle,'textbox',...
    [0.30 0.5 1 0.1],...
    'String',id_string,...
    'FontSize',40,...
    'FitBoxToText','off',...
    'EdgeColor','none');
   saveas(handle,'pie_temp.png','png');
   C = imread('pie_temp.png');
   C = imresize(C,0.2);

   red_matrix = get_red_matrix(raw);

   xCoord = round(xCoords(i)-x_scale/2+1);
   yCoord = round(yCoords(i)-y_scale/2+1);
   A(yCoord:(yCoord+y_scale-1),xCoord:(xCoord+x_scale-1),:) = C;
   if (i > 1)
       parent = get_parent(i,matrix);
       x1 = xCoords(i);
       y1 = yCoords(i);
       x2 = xCoords(parent);
       y2 = yCoords(parent);
       dist = sqrt((x2-x1)*(x2-x1)+(y2-y1)*(y2-y1));
       x1short = round(x1+(x2-x1)*(dist-70)/dist);
       x2short = round(x2+(x1-x2)*(dist-70)/dist);
       y1short = round(y1+(y2-y1)*(dist-70)/dist);
       y2short = round(y2+(y1-y2)*(dist-70)/dist);
       for k = 0:2
           pixels = drawLine(y1short,x1short+k,y2short,x2short+k);
           for j = 1:size(pixels,1)
               if(red_matrix(parent,i)==0)
                A(pixels(j,1),pixels(j,2),:) = [0,0,0];
               else
                A(pixels(j,1),pixels(j,2),:) = [255,0,0];
               end
           end
       end
   end
   close
end

[red_matrix,matrix_new,unique_new]=get_red_matrix(raw);
height_vec = get_height_vec_red(matrix_new,red_matrix);

xlswrite(filename,red_matrix,'red_matrix');
xlswrite(filename,matrix_new,'matrix_new');
xlswrite(filename,unique_new,'unique_new');


depth_vec = get_red_depth_vec(matrix_new,red_matrix);
report(:,8) = depth_vec;
report(:,9) = height_vec;
heading(1,1) = cellstr('Module_ID');
heading(1,2) = cellstr('Size');
heading(1,3) = cellstr('Cycle_Count');
heading(1,4) = cellstr('Cycle/Size');
heading(1,5) = cellstr('Num_Pies');
heading(1,6) = cellstr('Homogeneity');
heading(1,7) = cellstr('Dominance Factor');
heading(1,8) = cellstr('Depth');
heading(1,9) = cellstr('Height');

export(1,:) = heading;
export(2:size(report,1)+1,1:9) = num2cell(report);

xlswrite('Shred_Example_Data',export);



% image_string = strcat(sheetname,'.png');
image_file = strcat(filename,'.png');
imwrite(A,image_file,'png');
```

make_legend

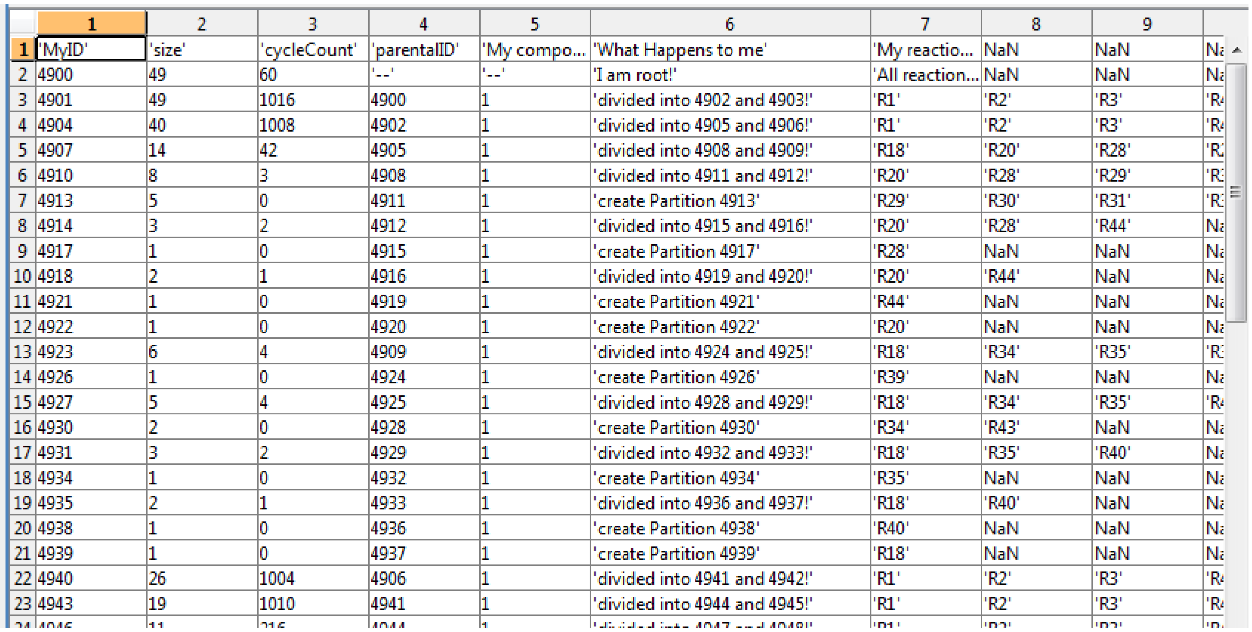
```
function make_legend(filename,sheetinfo)
% make a legend for the colors and what canonical metabolic pathways they
% represent.
% filename = string filename of the model
% The cell array of the color map uploaded from the model spreadsheet
% This function is called by Shred_Network()

vec = sheetinfo;

colormap=xlsread(filename,'Colormap');
```

```
num_groups = length(vec);
A = im2uint8(255*ones(num_groups*100+50,700,3));

for(i=1:num_groups)
figure1 = figure;

% Create textbox
annotation(figure1,'textbox',...
    [0.3 0.5 0.8 0.1],...
    'String',vec(i),...
    'FitBoxToText','off',...
    'FontSize',20,...
    'LineStyle','none');

handle = gcf;
saveas(handle,'test_text.png','png');
image = imread('test_text.png');
text_matrix = image(360:420,350:620,:);
var = 100*i-95;
A(var:var+60,5:275,:) = text_matrix;
imwrite(text_matrix,'text_matrix.png','png');
close
figure2 = figure;

% Create rectangle
annotation(figure1,'rectangle',...
    [0.3 0.5 0.2 0.05],...
    'FaceColor',colormap(i,:)/255);
    saveas(handle,'test_bar.png','png');
    image_2 = imread('test_bar.png');
    close
    bar_matrix = image_2(405:455,360:750,:);
    A(var:var+50,300:690,:)=bar_matrix;
    imwrite(bar_matrix,'bar_matrix.png','png');
end

imwrite(A,'Legend.png','png');
```

readDatafromLogFile

```
function [result] = readDataFromLogFile(file)

%reads in the rawoutput file Output.log produced by the partition funcitons
%and produces the raw data in a cell array format.
fid = fopen(file);
lines = textscan(fid,'%s','delimiter','\n');
fclose(fid);
lines = lines{1,1};
rows = size(lines,1);
columns = 0;
for r=1:rows
   line = lines{r,1};
   line = textscan(line, '%s','delimiter','\t');
   line = line{1,1};
   c = size(line,1);
   if columns < c
      columns = c;
   end
end
result = cell(rows,columns);
for r=1:rows
   line = lines{r,1};
   line = textscan(line, '%s','delimiter','\t');
   line = line{1,1};
   for c = 1:columns
      if c <= size(line,1)
         result{r,c} = line{c,1};
         [num, stat] = str2num(line{c,1});
         if stat == 1
            result{r,c} = num;
         end
      else
         result{r,c} = NaN;
      end
   end
end
```

The output from the log file (raw_data) looks like this:



	1	2	3	4	5	6	7	8	9	
1	'MyID'	'size'	'cycleCount'	'parentalID'	'My compo...	'What Happens to me'	'My reactio...	NaN	NaN	Na
2	4900	49	60	'--'	'--'	'I am root!'	'All reaction...	NaN	NaN	Na
3	4901	49	1016	4900	1	'divided into 4902 and 4903!'	'R1'	'R2'	'R3'	'R
4	4904	40	1008	4902	1	'divided into 4905 and 4906!'	'R1'	'R2'	'R3'	'R
5	4907	14	42	4905	1	'divided into 4908 and 4909!'	'R18'	'R20'	'R28'	'R
6	4910	8	3	4908	1	'divided into 4911 and 4912!'	'R20'	'R28'	'R29'	'R
7	4913	5	0	4911	1	'create Partition 4913'	'R29'	'R30'	'R31'	'R
8	4914	3	2	4912	1	'divided into 4915 and 4916!'	'R20'	'R28'	'R44'	Na
9	4917	1	0	4915	1	'create Partition 4917'	'R28'	NaN	NaN	Na
10	4918	2	1	4916	1	'divided into 4919 and 4920!'	'R20'	'R44'	NaN	Na
11	4921	1	0	4919	1	'create Partition 4921'	'R44'	NaN	NaN	Na
12	4922	1	0	4920	1	'create Partition 4922'	'R20'	NaN	NaN	Na
13	4923	6	4	4909	1	'divided into 4924 and 4925!'	'R18'	'R34'	'R35'	'R
14	4926	1	0	4924	1	'create Partition 4926'	'R39'	NaN	NaN	Na
15	4927	5	4	4925	1	'divided into 4928 and 4929!'	'R18'	'R34'	'R35'	'R
16	4930	2	0	4928	1	'create Partition 4930'	'R34'	'R43'	NaN	Na
17	4931	3	2	4929	1	'divided into 4932 and 4933!'	'R18'	'R35'	'R40'	Na
18	4934	1	0	4932	1	'create Partition 4934'	'R35'	NaN	NaN	Na
19	4935	2	1	4933	1	'divided into 4936 and 4937!'	'R18'	'R40'	NaN	Na
20	4938	1	0	4936	1	'create Partition 4938'	'R40'	NaN	NaN	Na
21	4939	1	0	4937	1	'create Partition 4939'	'R18'	NaN	NaN	Na
22	4940	26	1004	4906	1	'divided into 4941 and 4942!'	'R1'	'R2'	'R3'	'R
23	4943	19	1010	4941	1	'divided into 4944 and 4945!'	'R1'	'R2'	'R3'	'R
24	4946	11	216	4944	1	'divided into 4947 and 4948!'	'R1'	'R2'	'R3'	'R

Take module 4900, the parent id. An artificial module 4901 that holds all the reactions in 4900 is created, which is divided into 4902 and 4903. The get_results() and subsequent functions remove these artificial modules from the hierarchical tree of modules.

ShreddNewman_computeCompMatrix

```
% returns a reduced matrix, and a relevant mapper, based on the compnentID
% and the assignment of nodes in the componentIndex vector.

% if componentIndexVecotr = [1 3 5 1 1 1]
% and component ID is 1, then we will generate a 4x4 compMatrix with
% edges internal to that component


function [compMatrix, compMapper]=ShreddNewman_computeCompMatrix(matrix,
componentID,componentIndexVector) ;

% find rows and columns to eliminate
compMapper = find (componentIndexVector == componentID);

compMatrix = matrix (compMapper, compMapper);

end
```

ShreddNewman_get_Q_and_Eigenvector

```
% this function starts with the Newmans B matrix, and then computes
% the eigenvalues/vectors for the B matrix, and calculates the corresponding
Q.
%

function [Q, EigVector]= ShreddNewman_get_Q_andEigenVector (BMatrix)

    [V,D]=eig(BMatrix);
    EigVector= (V(1:size(BMatrix,1),size(BMatrix,1)))';   %#ok<FNDSB>
    Q_fid = fopen('get_Q.txt','w');
    fprintf(Q_fid,'interation\ti\tj\tQ\n');
    sMatrix =zeros(size(BMatrix, 1));
    Q=0;
    iteration =1;
    for i=1:size (BMatrix,1)
        for j=1:size (BMatrix,1)
            s_i = EigVector(i);
            s_j = EigVector(j);
            if((s_i <= 0 &&s_j <= 0)||(s_i >= 0&& s_j >= 0))
                s=1;
            else
                s=-1;
            end;
            sMatrix(i,j) = s;
            Q=Q+(BMatrix(i,j))*s;
            fprintf(Q_fid,'%f\t%f\t%f\t%f\n',iteration,i,j,Q);
            iteration = iteration+1;
        end;
    end;


end
```

ShreddNewman_partition

```
function [ShreddNewman_finalMap, ShreddNewman_finalMapHistory
]=ShreddNewman_partition(matrix, traditionalNewman, retroactivity)
  global ShreddNewman_fid;
  global ShreddNewmanModuleReport_fid;
  global ShreddNewmanModuleResults;
  global ShreddNewman_partitionID;
  global RC; %RC = Results - Cellarray
  global RI; % Row Index
  RI = 1;

  % set the initial partition ID.
  ShreddNewman_partitionID=  floor (100* size(matrix,1));
    % a global map to keep track of the mapping.  Initially every node is
    % in the "1" group.
    global ShreddNewman_finalMap;
    ShreddNewman_finalMap = ones(size(matrix,1),1);

    global  ShreddNewman_finalMapHistory ;
    ShreddNewman_finalMapHistory  = ShreddNewman_finalMap;

   ShreddNewmanModuleResults = [];

    % shows the history of the components.
%     global  ShreddNewman_componentHistory;
%     ShreddNewman_componentHistory= ones(size(matrix,1),1);

    % a nx1 matrix that marks the affiliations of nodes to their finest
module
    group=ones(size(matrix,1),1);

    % oldMapper maps from one level of the recursion to the next. Used when
    % returning from the recursive partitioning call. (at the top level
    % here it is not).
    oldMapper = ( 1:size(matrix,1))';
    recursionCounter = 0;



    fprintf(ShreddNewman_fid,'MyID\tsize\tcycleCount\tparentalID\tMy
component # within parent\tWhat Happens to me\tMy reactions... \n');
    fprintf(ShreddNewman_fid,'%d\t%d\t%d\t--\t--\tI am root!\tAll reactions
are included in Root!\n', ShreddNewman_partitionID, size(matrix,2),
util_CycleCount(matrix) );
    %util_CycleCount(matrix)
    ShreddNewman_partition_rec(recursionCounter,
matrix,matrix,group,ShreddNewman_partitionID,oldMapper, traditionalNewman,
retroactivity);
%     document the modules
%         Nodes Size    Cycles  Depth   Cycles/Size Is_Leaf_Node
Is_Homogeneous

fprintf(ShreddNewmanModuleReport_fid,'ModuleID\tSize\tCycleCount\tDepth\tCycl
es/Size\tIs_LeafNode\tIs_Homogeneous\n');
```

```
    for i = 1: size(ShreddNewmanModuleResults, 1)
        fprintf(ShreddNewmanModuleReport_fid,'%d\t%d\t%d\t%d\t%f\t%d\t%d\n',
ShreddNewmanModuleResults(i,1), ShreddNewmanModuleResults(i,2),
ShreddNewmanModuleResults(i,3),ShreddNewmanModuleResults(i,4),
ShreddNewmanModuleResults(i,5), ShreddNewmanModuleResults(i,6) ,
ShreddNewmanModuleResults(i,7));
    end

end
```

ShreddNewman_partition_rec

```
% recursive partitioning function
This is where the partitioning happens: The Q and the optimal s vector can be
computed many different ways, for example by GA, or eigenvector
approximation.

% February 4, 2011: modified to account for not partitioning when both
% children don't have cycles.

%  traditional Newman = not retroactive and not shredd.
%  Recursive partitioning:
% stopping critera:
% if traditionalnewman: single-node-componennt, or bad eigenvalues
% if ~traditionalnewman:  same as above, AND both children of partition do
% not have cycles.

function
[recursionCounter,group]=ShreddNewman_partition_rec(recursionCounter,
orig,matrix,group,parentPartitionID,oldMapper, traditionalNewman,
retroactivity)
global ShreddNewman_finalMap;
global ShreddNewman_fid;
global ShreddNewman_reactionData;
global ShreddNewmanModuleResults;
global ShreddNewman_partitionID;

%Begin Gautham 03/13/2011 11:45pm
global ave_std_shredd_fid
%End Gautham 03/13/2011 11:45pm




cyclesDontMatter = traditionalNewman;

[ numConnectedComponents, CCsize, componentIndexVector] =
util_getComponents(matrix);
%
% fprintf(ShreddNewman_fid,'\n');
% util_printIndents(ShreddNewman_fid, recursionCounter);
% fprintf(ShreddNewman_fid,'RC=%d PartitionID=%d matrixSize=%d
Componnts=%d\n', recursionCounter, parentPartitionID, size(matrix,2),
numConnectedComponents );
```

```
% if a module with more than one component, then write it now -- as it will
% be made into seperate components

ModuleHasHomogenousComposition = true;

if (numConnectedComponents > 1)

    tempIndexVector = ones (1,size(matrix,2));
    [ModuleMatrix, ModuleMapper] = ShreddNewman_computeCompMatrix(matrix,
1,tempIndexVector) ;
    % fprintf(ShreddNewman_fid,'MyID\tsize\tcycleCount\tparentalID\tMy
component # within parent\t\What Happens to me\n');
    fprintf(ShreddNewman_fid,'%d\t%d\t%d\t%d\t0\tbecomes %d copmponents\t',
parentPartitionID, size(matrix,2), util_cycleCount (matrix)
,parentPartitionID,  numConnectedComponents );
    % make me an official partition and print all my reactions
     %util_cycleCount (matrix)

    ModuleHasHomogenousComposition = true;

    oldComposition = util_findComposition(oldMapper(ModuleMapper(1)));

    for x=1:size(ModuleMatrix,1)

        text = sprintf('%s',
char(ShreddNewman_reactionData(oldMapper(ModuleMapper(x)))));
        fprintf(ShreddNewman_fid,'%s\t', char(text));
        newComposition  = util_findComposition( oldMapper(ModuleMapper(x)));
        if (~strcmp(char(oldComposition), char(newComposition) ))
            ModuleHasHomogenousComposition= false;
        end
    end
    fprintf(ShreddNewman_fid,'\n');

end

ModuleCycleCount = 0;

for componentID=1:numConnectedComponents
    %Gautham 03142986
    test_index = 1;
    % Make the component Matrix/Mapper
    [compMatrix, compMapper] = ShreddNewman_computeCompMatrix(matrix,
componentID,componentIndexVector) ;

    %% 6 lines below on 11/11/2011 at 420pm
    CC_matrix = zeros(size(compMatrix,1),size(compMatrix,1));
    for(i=1:size(compMatrix,1))
        for(j=1:size(compMatrix,2))
            if(compMatrix(i,j) ~=0)
                CC_matrix(i,j) = 1;
            end
        end
    end
    localCycleCount = util_cycleCount (CC_matrix);
%     util_cycleCount (compMatrix);
    ModuleCycleCount = ModuleCycleCount + localCycleCount; %indivdiual sum of
cycles in each component make up the sume of cycles in the current Module.
    componentHasHomogenousComposition= true;
    componentIsLeaf = false;
    oldComposition = util_findComposition( oldMapper(compMapper(1)));  %
composition of first member of the group



    % each component gets a new ID:
    ShreddNewman_partitionID= ShreddNewman_partitionID+1;
    component_partitionID = ShreddNewman_partitionID;

    % initialize  conditions:

    Q=-1;
    part_a_has_cycle =0; part_b_has_cycle =0;
    mixedEValues =0;




    if CCsize(componentID) > 1
        % stopping critera:
        % if traditionalnewman: single-node-componennt, or bad eigenvalues
        % if ~traditionalnewman:  same as above, AND both children of
        % partition do not have cycles.

        % reduce the matrix based on componnent ID and nodes belonging
        % to the component:

        hasCycles = util_hasCycle (compMatrix);
        part_a_has_cycle =0; part_b_has_cycle =0;

        if (hasCycles || cyclesDontMatter)
            %Gautham 03/14/2011 at 12:05am - changed the output of the
            %function below
            [BMatrix,ave_shred,std_shred,num_shred] =
ShreddNewman_populate_BMatrix2(compMatrix, traditionalNewman, retroactivity);
            fprintf(ave_std_shredd_fid,'%d\t%d\t%d\t%d
\n',component_partitionID,ave_shred,std_shred,num_shred);
            %compute Q & eigenvalues
%             [~,~,EigVector,Q]=test_split(compMatrix,BMatrix);
            % Below commented out 07/03/2011 230pm

            %09/08/11
%             [Q,EigVector] = GA_s_vector(BMatrix,compMatrix);
            % 08/26/11
%             [Q,EigVector] = get_MINLP_s(BMatrix);
%              [Q, EigVector] = ShreddNewman_get_Q_andEigenVector (BMatrix);
             %02/15/2012
             [Q,EigVector] = GA2_s_vector(BMatrix,compMatrix);
             %Changed from size(x,2) to length() because of the Eigenvector
             %could be a column vector. 12/02/2012 at 11:34pm.

            mixedEValues = length (find(EigVector>=0)) > 0 && length
(find(EigVector<0)) > 0;
        end

        if ((cyclesDontMatter || hasCycles) && mixedEValues && Q>0.1)


            % process the left partition first:
            ShreddNewman_partitionID= ShreddNewman_partitionID+1;
            leftPartitionID = ShreddNewman_partitionID;



            for i=1:size(compMatrix,1);
                s_i = EigVector(i);
                % take into account the compMapper!!
                if(s_i< 0.0000000000001)
                    group(oldMapper(compMapper(i)))=leftPartitionID;
                end
            end


[part_matrix_a,mapper_a]=ShreddNewman_split(orig,leftPartitionID,group,oldMap
per);
            part_a_has_cycle = util_hasCycle (part_matrix_a);

            % process the right partition
            ShreddNewman_partitionID= ShreddNewman_partitionID+1;
            rightPartitionID = ShreddNewman_partitionID;

            for i=1:size(compMatrix,1);
                s_i = EigVector(i);
                % take into account the compMapper!!
                if(s_i >=0.0000000000001)
                    group(oldMapper(compMapper(i)))=rightPartitionID;
                end
            end
            [part_matrix_b,mapper_b
]=ShreddNewman_split(orig,rightPartitionID,group,oldMapper);
            part_b_has_cycle = util_hasCycle (part_matrix_b);

            % keep the two lines below if you don't want the cycle count
            % check. Edited 11/07/11 at 10:36pm

            part_a_has_cycle = 1;
            part_b_has_cycle = 1;

            if (part_a_has_cycle && part_b_has_cycle)

                %
fprintf(ShreddNewman_fid,'MyID\tsize\tcycleCount\tparentalID\tMy component #
within parent\t\What Happens to me\n');
```

```
                fprintf(ShreddNewman_fid,'%d\t%d\t%d\t%d\t%d\tdivided into %d
and %d!\t', component_partitionID, size(compMatrix,2),
localCycleCount,parentPartitionID, componentID, leftPartitionID,
rightPartitionID );
                % make me an official partition and print all my reactions

                for x=1:size(compMatrix,1)
                    text = sprintf('%s',
char(ShreddNewman_reactionData(oldMapper(compMapper(x)))));
                    fprintf(ShreddNewman_fid,'%s\t', char(text));
                    newComposition  = util_findComposition(
oldMapper(compMapper(x)));
                    if (~strcmp(char(oldComposition), char(newComposition) ))
                        componentHasHomogenousComposition= false;
                    end
                end
                fprintf(ShreddNewman_fid,'\n');
                % recursively call paritioning on mychildren:
                recursionCounter = recursionCounter+1;
                [recursionCounter, Output_2]
=ShreddNewman_partition_rec(recursionCounter,orig,part_matrix_a,group,leftPar
titionID,mapper_a, traditionalNewman, retroactivity);
                [recursionCounter,
Output_2]=ShreddNewman_partition_rec(recursionCounter,orig,part_matrix_b,grou
p,rightPartitionID,mapper_b, traditionalNewman, retroactivity);
                recursionCounter = recursionCounter-1;
            end

        end

    end

    if (~part_a_has_cycle || ~part_b_has_cycle)
        componentHasHomogenousComposition = true;
        componentIsLeaf = true;
        for i=1:size(compMatrix,1);

ShreddNewman_finalMap(oldMapper(compMapper(i)))=component_partitionID;
        end
        fprintf(ShreddNewman_fid,'%d\t%d\t%d\t%d\t%d\tcreate Partition %d\t',
component_partitionID, size(compMatrix,2),
localCycleCount,parentPartitionID, componentID, component_partitionID );
        % print all my reactions
        for x=1:size(compMatrix,1)
            text = sprintf('%s',
char(ShreddNewman_reactionData(oldMapper(compMapper(x)))));
            fprintf(ShreddNewman_fid,'%s\t', char(text));
            newComposition  = util_findComposition(
oldMapper(compMapper(x)));
            if (~strcmp(char(oldComposition), char(newComposition) ))
                componentHasHomogenousComposition= false;
            end
        end
        fprintf(ShreddNewman_fid,'\n');
    end


%'ModuleID\tSize\tCycleCount\tDepth\tCycles/Size\tIs_LeafNode\tIs_Homogeneous
\n');
    ModuleInfo = [  component_partitionID size(compMatrix,1) localCycleCount
recursionCounter  localCycleCount/size(compMatrix,1)  componentIsLeaf
componentHasHomogenousComposition];
    ShreddNewmanModuleResults = [ShreddNewmanModuleResults; ModuleInfo];
%#ok<AGROW>



    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    % document the parent component if we had multipe components:
    if (componentID == numConnectedComponents && componentID > 1)
        ModuleInfo = [ parentPartitionID  size(ModuleMatrix,1)
ModuleCycleCount  recursionCounter  ModuleCycleCount/size(ModuleMatrix,1)  0
ModuleHasHomogenousComposition];
        ShreddNewmanModuleResults = [ShreddNewmanModuleResults; ModuleInfo];
%#ok<AGROW>
    end



end
%Gautham 03/14/2011 at 11:52am
% fclose(ave_std_shredd_fid)
%End Gautham at 03/14/2011 11:52am
end
```

ShreddNewman_populate_Bmatrix

```
%key idea here is to compute a Bmatrix, a symmetric matrix, This is the same
as the G-matrix in Chapter 2 and the V matrix in Chapter 3. The computation
of V is default, and the computation based on arithmetic average and expected
ShReD is commented out.

%Gautham 03/13/2011 - Changed the output of the matrix to hold the average
%shred of the module as well as the standard deviation of the shreds.

function
[bmatrix,ave_shred,std_shred,num_shred]=ShreddNewman_populate_BMatrix2(adjmat
rix, traditionalNewman, retroactivity)

%Begin Gautham 03/13/2011 11:49pm
%initialized to 0 because we won't always be dealing with Shreds so it
%should output 0 by default.
global ave_std_shredd_fid;
global network_size;
global filename;

ave_shred = 0;
std_shred = 0;
num_shred = 0;
```

```
bmatrix = zeros (size(adjmatrix,1));

% doing directional newman algorithm
if (traditionalNewman)
    %  disp (adjmatrix);

    m  = sum(sum(adjmatrix));
    if (m ~= 0)
        for i=1:size(adjmatrix,1)
            outdegreeI = sum(adjmatrix(i, 1:size(adjmatrix,1)));
            for j=1:size(adjmatrix,1);
                indegreeJ = sum(    adjmatrix(1:size (adjmatrix,1), j) );
                % adjmatrix - (outdegree of i)*(indegree of j) / 2m
                bmatrix(i,j) =  adjmatrix (i,j) - ( outdegreeI * indegreeJ /
(2*m));

            end;
        end;
    end
    % make it summetrical to account for the difference between
    % connectivity (i to j and j to i) and the expected # of connectsion
    % between i and j and the # of connectsion between j and i
    % BM (i,j) = #conns (i,j) - expecte # of conns for i,j
    bmatrix = bmatrix + bmatrix';

else

    FWdistance = floyd_warshall_all_sp (sparse(adjmatrix));
    shredd = FWdistance + FWdistance';
      shredd(shredd>10000)=10000;

    % for all distances greater than 1, replace with very large numbers! if
    % doing local retroactivity:

    if( retroactivity)
        shredd(shredd > 1) = 100000;
    end

 fid_dist = fopen('dist.txt','w');
 fid_P_ij = fopen('expected_actual.txt','w');
 fprintf(fid_P_ij,'Iteration:\ti\tj\tShred_ij\tExpected Shred\n');
 iteration = 1;
    for i=2:size(shredd,1);
        for j=1:i-1;
            fprintf('Second Loop: i:%ij;%i\n',i,j)
            row_i = shredd(i,:);
            row_i([i,j]) = [];
            row_j = shredd(j,:);
            row_j(j)=[];
            shred_dist = sort([row_i,row_j]);
            index = bsearch(shred_dist,shredd(i,j));
```

```
%                  expected_shred_dist =
get_expected_shred_dist(FWdistance,i,j);
%                  p =
sum(expected_shred_dist>=shredd(i,j))/length(expected_shred_dist);
%Below 07042011

%08/18/11
%Find all the shreds out of i, then the ones out of j. then get a
%distribution of shreds that only involvei and j. Find p which is the
%probability that a randomly selected shred will be greater than the
%shred_ij, then calculate b as log(p/1-p)

        p=(length(shred_dist)-index)/length(shred_dist);
        %consider making all large shreds have small p
        if(p==0)
            p = 0.01;
        end
        if(p==1)
            p = 0.99;
        end

        bmatrix(i,j) = log(p/(1-p));
        bmatrix(j,i) = bmatrix(i,j);
%                      bmatrix(i,j) = p-.5;
        if(i==j)
            bmatrix(i,j)=0;
        end
%                      test = 5;

%                      End 07042011
%Do line below for normal old bmatrix, and comment out stuff above.
%                 bmatrix(i,j) = -shredd (i,j) + (avgShredd(i) +
avgShredd(j))/2;
%
fprintf(fid_P_ij,'%f\t%f\t%f\t%f\t%f\n',iteration,i,j,shredd(i,j),(avgShredd(
i)+avgShredd(j))/2);
%              end;
        end;
    end;

    bmatrix_made = 1
    size(bmatrix,1)
    fclose(fid_P_ij);
    % zero the diagonal for shredd based analysis
    % more efficient than checking to make sure not to subtract the avgs
above.
    bmatrix = bmatrix - diag (diag(bmatrix));
    fclose(fid_dist);
%     xlswrite('Shredd_dump.xlsx',bmatrix,'bmatrix');
end

if(size(bmatrix,1)==network_size)
    xlswrite(filename,bmatrix,'Mod_Matrix');
    xlswrite(filename,shredd,'Shred_Matrix');
```

```
end
end
```

ShreddNewman_run

```
function ShreddNewman_run(inputFile, inputSheet, regulationSheet, outputFile,
traditionalNewman, retroactivity, includeRegulation, include_flux,fluxes)
% Takes in stochiometric matrix in the input sheet of the inputFile
% creates a directional adjacnecy matrix
% calls ShreddNewman on it.
%
% inputFile - filename of .xlsx spreadsheet with adjacency matrix
% inputSheet - sheet in .xlsx file containing matrix
% outputFile - base filename for all outputfiles.
% traditionalNewman: if true: runs the traditional newman method
% retroactivity: if true, run local retroactivity where all distances are
% set to 1
% if both traditionalNew and retroactivity are false, then run Newman using
% shredd metric

global ShreddNewman_fid;
global ShreddNewmanModuleReport_fid;

global ShreddNewman_reactionData;

global ShreddNewmanCompositionLegend;  %composition legend
global compare_GA_eig_fid;
global filename;

compare_GA_eig_fid = fopen('compare_GA_eig.txt','w+');
fprintf(compare_GA_eig_fid,'Module_Size\t Q_GA\t Q_eig\n');

%Gautham 03/14/2011 at 1:52pm
global ave_std_shredd_fid;
ave_std_shredd_fid = fopen('ave_std_shredd.txt','w');
fprintf(ave_std_shredd_fid,'Module Id \t Average_Shred \t
Standard_Deviation_Shred \t Num_Shred \n');
%End Gautham 03/14/2011 at 1:52pm.


ShreddNewman_fid = fopen(strcat(outputFile, '.log'), 'w+');
ShreddNewmanModuleReport_fid = fopen(strcat(outputFile, '_ModuleReport'),
'w+');
% read in spreadsheet and generate a reaction graph

if(include_flux)

[recMatrix,interaction_matrix,ShreddNewman_reactionData,Output_4]=get_rxn_mat
rix_4(inputFile, inputSheet, fluxes, regulationSheet, includeRegulation);
else
```

```
    [recMatrix, ShreddNewman_reactionData, Output_3] =
util_WriteReactionMatrixForCytoscape(inputFile, inputSheet, regulationSheet,
includeRegulation);
end

xlswrite(filename,recMatrix,'recMatrix');

[Output_1,Output_2,legend_raw]=xlsread(inputFile,'Classifications');
ShreddNewmanCompositionLegend = legend_raw(:,2);

ShreddNewman_partition(recMatrix, traditionalNewman, retroactivity);


fclose (ShreddNewman_fid );
fclose (ShreddNewmanModuleReport_fid );
fclose(ave_std_shredd_fid);
fclose(compare_GA_eig_fid);

% write out the reaction matrix in adjacency list format.
if (includeRegulation)
    ram_fid = fopen(strcat(inputFile, '_ReactionAdjacencyMatrixWithReg'),
'w+');
else
       ram_fid = fopen(strcat(inputFile, '_ReactionAdjacencyMatrix'), 'w+');
end
util_writeAdjMatrix( recMatrix, ShreddNewman_reactionData, ram_fid );

end
```

ShreddNewman_split

```
%splits a partition matrix from the original one.

% the part_matrix will be an n x n matrix of the size of the partition.
% the mapper is a mapping from from the NEW nodes numbers (1... n) to the
% old node numbers (i.e. if mapper is 3 5 7, then node 0 of this partition
% is the older node 3, and node 1 of this partition is the old node 5, nad
% so on

function [part_matrix,mapper] =
ShreddNewman_split(orig,partition,group,oldMapper)




mapper = zeros(size(group)); %keep track of remappings
partCount = 0; %number of nodes in new partition


for i=1:size(oldMapper, 1)
    if (group (oldMapper(i)) == partition)
        partCount = partCount+1;
        mapper (partCount) = oldMapper(i);
    end;
```

```
end;
mapper(mapper == 0) = [];


%making the new adjacency matrix for the first partition
part_matrix = zeros(partCount,partCount);
for j=1:size(mapper);
    for i=1:size(mapper);
        %disp('split values')
        %disp(orig(mapper(i),mapper(j)))
        part_matrix(i,j)=orig(mapper(i),mapper(j));
        part_matrix(j,i)=orig(mapper(j),mapper(i));
    end;
end;
```

util_cycleCount

input: Adjacency matrix for directed graph

output: Number of directed cycles in the graph. This is approximate for larger graphs, since run time is factoral in the size of the adjacency matrix.

util_getComponents

```
function[ componentNum, componentLengths, nodeComponents] =
util_getComponents(adjMatrix)

% Takes in an adjacency matrix and returns the number of connected components
in the graph, a
% vector of the lengths of all components , and a vector identifying the
component
% of each node (i.e. if nodeComponents(i) == 5, node i is in component 5)

    numNodes = length(adjMatrix);

    nodeComponents(1:numNodes) = 0;
    visited(1:numNodes) = 0;

    componentNum = 0;
    componentLengths = [];

    while min(visited) == 0
        thisComponentLength = 0;
        componentNum = componentNum + 1;
        [value, node] = min(visited);
        to_visit = node;

        while numel(to_visit) ~= 0
            currentNode = to_visit(1);
            visited(currentNode) = 1;
            to_visit(1) = [];
            for i = 1:numNodes
                if (adjMatrix(currentNode, i) >= 1 || adjMatrix(i,
currentNode) >= 1) && visited(i) == 0
                    to_visit = [to_visit i];
                    visited(i) = 1;
                end
            end
        end

        for i = 1:numNodes
            if visited(i) == 1
                nodeComponents(i) = componentNum;
                thisComponentLength = thisComponentLength + 1;
                visited(i) = 2;
            end
        end

        %if thisComponentLength > 1
            componentLengths = [componentLengths thisComponentLength];
        %end
    end

end
```

util_hasCycle

input: Adjacency matrix for directed graph

output: graph has at least one directed cycle

util_writeReactionMatrixForCytoscape

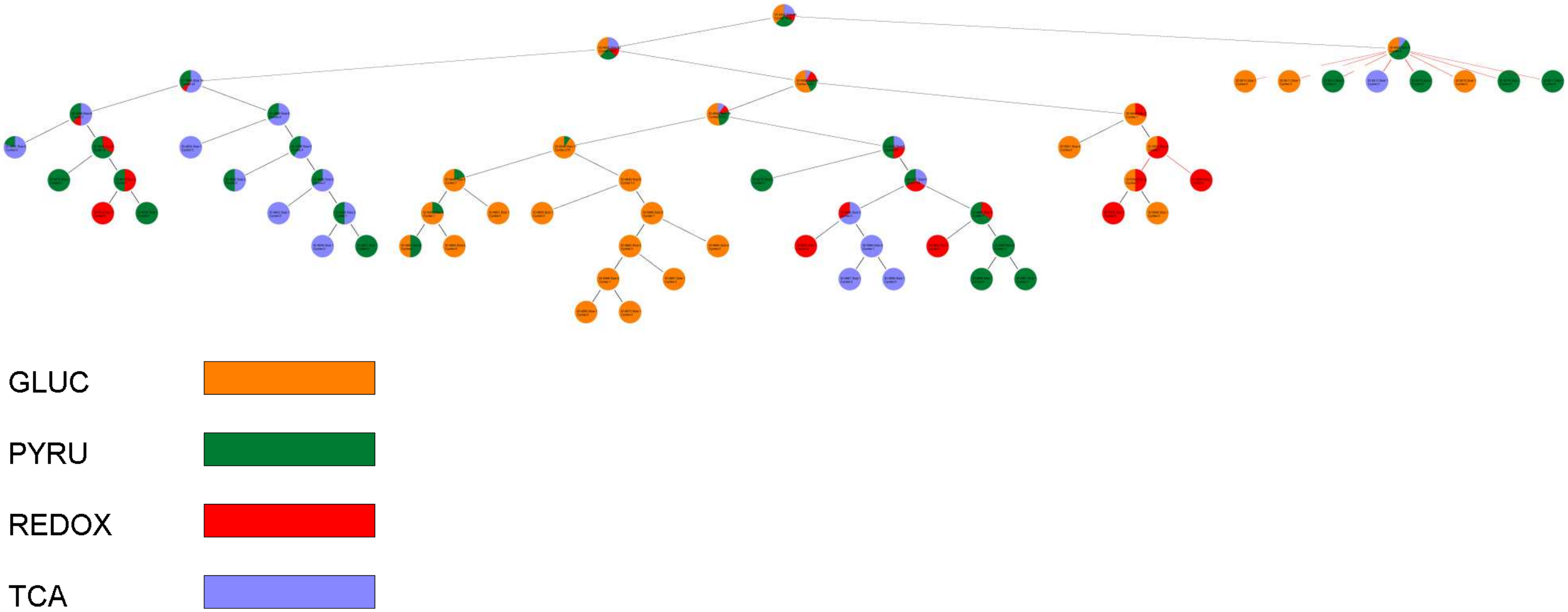
Get adjacency matrix if no flux weights are provided. Called by ShreddNewman_run

```
function [ ReactionAdjMatrix, ReactionLabels,  numReactions ]
=util_WriteReactionMatrixForCytoscape( inputFile, inputSheet,
regulationSheet, regulation)


r =regexp (inputFile, '\.', 'split');
if (regulation)
    s= strcat(r(1), '_reactionRegCyto.txt' );
else
    s= strcat(r(1), '_reactionCyto.txt' );
end
fid = fopen(char(s(1)), 'w+');

% read from xls
[numData, textData] = xlsread(inputFile, inputSheet);


reversibility = numData(1,:);
adjMatrix = numData(3:end, :);
numMetabolites = size (adjMatrix,1);
numReactions = size(adjMatrix, 2);
ReactionLabels(1:numReactions) = textData(2,3:end);

formattedTextData(1:numMetabolites) = textData(3:end,2);
formattedTextData(numMetabolites + 1:numMetabolites + numReactions) =
textData(2,3:end);

ReactionAdjMatrix = zeros(numReactions);


for i = 1: numReactions
    column =  adjMatrix (:, i);
%     assert (sum (column < 0) > 0); % at least one negative entry in the
column

    outgoingMetabolites = find ((column > 0));



    for j = 1: size (outgoingMetabolites, 1)
        % find -1 in the row in outgoingMetabolites (j)
        row = adjMatrix (outgoingMetabolites (j), :);
        for k = 1: size (row, 2)
            %if ((row(k) == -1) && (i ~=k))
            if ((row(k) < 0) && (i ~=k))
                ReactionAdjMatrix (i, k) = 1;

fprintf(fid,'%s\tRR.\t%s\n',char(formattedTextData(i+numMetabolites)),char(fo
rmattedTextData(k+numMetabolites)));
                %   fprintf(fid,'m%s\t-->\tR%s\t-->m%s\t--
>\tR%s\n',char(formattedTextData(incomingMetabolites(z))),char(formattedTextD
ata(i+numMetabolites)),
char(formattedTextData(outgoingMetabolites(j))),char(formattedTextData(k+numM
etabolites));
            end

            if (reversibility (k))
                if ((row(k) > 0) && (i~=k) )
                    ReactionAdjMatrix (i, k) = 1;

fprintf(fid,'%s\tRR*\t%s\n',char(formattedTextData(i+numMetabolites)),char(fo
rmattedTextData(k+numMetabolites)));
                end

            end
        end

    end



    if (reversibility (i) == 1)

        incomingMetabolites = find ((column < 0));

        for z = 1:size (incomingMetabolites, 1)
            row = adjMatrix (incomingMetabolites (z), :);
            for k = 1: size (row, 2)
```

```
                %05/26/2011 removed the && reversibility(k) in the line
                %below.
                if ((row(k) < 0) && (i ~=k))
                    ReactionAdjMatrix (i, k) = 1;
                    fprintf(fid,'%s\tRR--
R\t%s\n',char(formattedTextData(i+numMetabolites)),char(formattedTextData(k+n
umMetabolites)));
                end
                if (reversibility (k))
                    if ((row(k) > 0) && (i~=k) )
                        ReactionAdjMatrix (i, k) = 1;

fprintf(fid,'%s\tRR*\t%s\n',char(formattedTextData(i+numMetabolites)),char(fo
rmattedTextData(k+numMetabolites)));
                    end

                end
            end
        end
    end

end


if(regulation)
    % add  regulation to reaction network

    [regnumData, regTextData] = xlsread(inputFile, regulationSheet);

    % assert all reactions are specified as in the non-regulation sheet
    regLabels = regTextData(1,3:end);

    reverseAdjMatrix= zeros(size(adjMatrix, 1), size (adjMatrix,2));
    %create a reverse adjMatrix
    for i= 1: numReactions
        if (reversibility (i) == 1)
            reverseAdjMatrix(:,i) = -adjMatrix(:,i);
        end

    end



    for i= 1: numReactions

        column =  regnumData (:, i);
        MetabolitesRegulatingReaction =    find ((column ~= 0));


        for metaboliteJ = 1: size(MetabolitesRegulatingReaction,1)
            % find all reactions producing metabolite metaboliteJ. look in
adjMatrix
```

```
            reactionsProducingMetaboliteJ = find
(adjMatrix(MetabolitesRegulatingReaction(metaboliteJ), :) > 0);
            additional = find
(reverseAdjMatrix(MetabolitesRegulatingReaction(metaboliteJ), :) > 0);
            reactionsProducingMetaboliteJ = [ reactionsProducingMetaboliteJ,
additional]; %#ok<AGROW>

            for z=1: size (reactionsProducingMetaboliteJ, 2)
                ReactionAdjMatrix (reactionsProducingMetaboliteJ(z),i) = 1;
                fprintf(fid,'%s\trREG\t%s\n',
char(formattedTextData(numMetabolites+reactionsProducingMetaboliteJ(z))),
char(formattedTextData(i+numMetabolites)));
            end

        end

    end
end


ReactionAdjMatrix = ReactionAdjMatrix  - diag (diag(ReactionAdjMatrix));


fclose(fid);

end
```

Final Output Hierarchical Tree of Modules



Bibliography

Acerenza,L. and Ortega,F. (2007) Modular metabolic control analysis of large responses The general case for two modules and one linking intermediate. **274**, 188–201.

Alves,R. *et al.* (2008) Mathematical formalisms based on approximated kinetic representations for modeling genetic and metabolic pathways. *Biotechnology & genetic engineering reviews*, **25**, 1–40.

Andrianantoandro,E. *et al.* (2006) Synthetic biology: new engineering rules for an emerging discipline. *Molecular systems biology*, **2**, 2006.0028.

Antunes,L.C.M. *et al.* (2011) Effect of antibiotic treatment on the intestinal metabolome. *Antimicrob Agents Chemother.*, **55**, 1494–503.

Aziz,R.K. *et al.* (2012) SEED servers: high-performance access to the SEED genomes, annotations, and metabolic models. *PloS one*, **7**, e48053.

Bailey,S.W. and Ayling,J.E. (2009) The extremely slow and variable activity of dihydrofolate reductase in human liver and its implications for high folic acid intake. *Proceedings of the National Academy of Sciences of the United States of America*, **106**, 15424–9.

Bajad,S.U. *et al.* (2006) Separation and quantitation of water soluble cellular metabolites by hydrophilic interaction chromatography-tandem mass spectrometry. *J Chromatogr A.*, **1125**, 76–88.

Bansal,T. *et al.* (2010) The bacterial signal indole increases epithelial-cell tight-junction resistance and attenuates indicators of inflammation. *Proc Natl Acad Sci USA*, **107**, 228–33.

Barrett,C.L. *et al.* (2009) Decomposing complex reaction networks using random sampling, principal component analysis and basis rotation. *BMC systems biology*, **3**, 30.

Beck,S. a and Tisdale,M J (2004) Effect of cancer cachexia on triacylglycerol/fatty acid substrate cycling in white adipose tissue. *Lipids*, **39**, 1187–9.

Bjeldanes,L.F. *et al.* (1991) Aromatic hydrocarbon responsiveness-receptor agonists generated from indole-3-carbinol in vitro and in vivo: comparisons with 2,3,7,8-tetrachlorodibenzo-p-dioxin. *Proc Natl Acad Sci USA*, **88**, 9543–7.

Blank,L.M. *et al.* (2005) Large-scale 13C-flux analysis reveals mechanistic principles of metabolic network robustness to null mutations in yeast. *Genome biology*, **6**, R49.

Boghigian,B. a *et al.* (2010) Metabolic flux analysis and pharmaceutical production. *Metabolic engineering*, **12**, 81–95.

Brown,M. *et al.* (2011) Automated workflows for accurate mass-based putative metabolite identification in LC/MS-derived metabolomic datasets. *Bioinformatics*, **27**, 1108–12.

Burcelin,R. *et al.* (2011) Gut microbiota and diabetes: from pathogenesis to therapeutic perspective. *Acta Diabetol.*, **48**, 257–73.

Carlisle-Moore,L. *et al.* (2005) Substrate recognition by the human fatty-acid synthase. *The Journal of biological chemistry*, **280**, 42612–8.

Chalhoub,E. *et al.* (2007) A computer model of gluconeogenesis and lipid metabolism in the perfused liver. *American journal of physiology. Endocrinology and metabolism*, **293**, E1676–86.

Chassaing,B. and Darfeuille-Michaud,A. (2011) The commensal microbiota and enteropathogens in the pathogenesis of inflammatory bowel diseases. *Gastroenterology*, **140**, 1720–28.

Chou,H.-H. *et al.* (2002) Inactivation of CMP-N-acetylneuraminic acid hydroxylase occurred prior to brain expansion during human evolution. *Proceedings of the National Academy of Sciences of the United States of America*, **99**, 11736–41.

Cormen,T.H. *et al.* (2001) Introduction to algorithms MIT press.

Croes,D. *et al.* (2006) Inferring meaningful pathways in weighted metabolic networks. *Journal of molecular biology*, **356**, 222–36.

Dervartanian,D.V. and Veeger,C. (1964) Studies on succinate dehydrogenase: I. Spectral properties of the purified enzyme and formation of enzyme-competitive inhibitor complexes. *Biochimica et biophysica acta*, **92**, 233–247.

Dumas,M.-E. *et al.* (2006) Metabolic profiling reveals a contribution of gut microbiota to fatty liver phenotype in insulin-resistant mice. *Proc Natl Acad Sci U S A*, **103**, 12511–6.

Van Duynhoven,J. *et al.* (2011) Metabolic fate of polyphenols in the human superorganism. *Proc Natl Acad Sci USA*, **108**, 4531–8.

Ederer,M. *et al.* (2003) An Approach for Dividing Models of Biological Reaction Networks into Functional Units. *Simulation*, **79**, 703–716.

Fell,D.A. (1992) Metabolic Control Analysis : experimental development of its theoretical and. *Biochemical Journal*, **286**, 313–330.

Floyd,R.W. (1962) Algorithm-97-Shortest Path. *Communications of the ACM*, **5**, 345.

Gebauer,J. *et al.* (2012) Detecting and investigating substrate cycles in a genome-scale human metabolic network. *The FEBS journal*, **279**, 3192–202.

Gille,C. *et al.* (2010) HepatoNet1: a comprehensive metabolic reconstruction of the human hepatocyte for the analysis of liver physiology. *Molecular systems biology*, **6**, 411.

Girvan,M. and Newman,M.E.J. (2002) Community structure in social and biological networks. *Proc Natl Acad Sci U S A*, **99**.

Goeddel,D. V *et al.* (1979) Expression in Escherichia coli of chemically synthesized genes for human insulin. *Proceedings of the National Academy of Sciences of the United States of America*, **76**, 106–10.

Goemann,B. *et al.* (2011) Topological peculiarities of mammalian networks with different functionalities : transcription , signal transduction and metabolic networks. *Network*, **1**, 134–148.

Greenblum,S. *et al.* (2011) Metagenomic systems biology of the human gut microbiome reveals topological shifts associated with obesity and inflammatory bowel disease. *Proc Natl Acad Sci USA*, **109**, 594–9.

Grimbs,S. *et al.* (2007) The stability and robustness of metabolic states: identifying stabilizing sites in metabolic networks. *Molecular systems biology*, **3**, 146.

Grochow,J.A. and Kellis,M. (2007) Network Motif Discovery Using Subgraph Enumeration and Symmetry-Breaking. 92–106.

Grosse,I. *et al.* (2009) GI-Edition Bioinformatics 2009.

Guan,H.-P. *et al.* (2002) A futile metabolic cycle activated in adipocytes by antidiabetic agents. *Nature medicine*, **8**, 1122–8.

Gutteridge,A. *et al.* (2007) Regulation of metabolic networks by small molecule metabolites. *BMC bioinformatics*, **8**, 88.

Handelsman,J. (2004) Metagenomics : Application of Genomics to Uncultured Microorganisms. *Microbiol Mol Biol Rev.*, **68**, 669–85.

Hartwell,L.H. *et al.* (1999) From molecular to modular cell biology. *Nature*, **402**, C47–52.

Heath-Pagliuso,S. *et al.* (1998) Activation of the Ah receptor by tryptophan and tryptophan metabolites. *Biochemistry*, **37**, 11508–15.

Heijnen,J.J. (2005) Approximative kinetic formats used in metabolic network modeling. *Biotechnology and bioengineering*, **91**, 534–45.

Heinemann,M. and Panke,S. (2006) Synthetic biology--putting engineering into biology. *Bioinformatics (Oxford, England)*, **22**, 2790–9.

Holme,P. *et al.* (2003) Subnetwork hierarchies of biochemical pathways. *Bioinformatics*, **19**, 532–538.

Hoppe,A. *et al.* (2007) Including metabolite concentrations into flux balance analysis: thermodynamic realizability as a constraint on flux distributions in metabolic networks. *BMC systems biology*, **1**, 23.

Hung,D.T. *et al.* (2005) Small-molecule inhibitor of Vibrio cholerae virulence and intestinal colonization. *Science (New York, N.Y.)*, **310**, 670–4.

Ideker,T. and Krogan,N.J. (2012) Differential network biology. *Molecular systems biology*, **8**, 565.

Ihmels,J. *et al.* (2004) Principles of transcriptional control in the metabolic network of Saccharomyces cerevisiae. *Nature biotechnology*, **22**, 86–92.

Jamshidi,N. and Palsson,B.Ø. (2008) Formulating genome-scale kinetic models in the post-genome era. *Molecular systems biology*.

Jeong,H *et al.* (2000) The large-scale organization of metabolic networks. *Nature*, **407**, 651–4.

Johnson,D.B. (1977) Efficient Algorithms for Shortest Paths in Sparse Networks. *Journal of the ACM*, **24**, 1–13.

Jones,B. V *et al.* (2008) Functional and comparative metagenomic analysis of bile salt hydrolase activity in the human gut microbiome. *Proc Natl Acad Sci USA*, **105**, 13580–5.

Kahn,A.B. (1962) Topological Sorting of Large Networks. *Communications of the ACM*, 668–562.

Kanehisa,Minoru *et al.* (2006) From genomics to chemical genomics: new developments in KEGG. *Nucleic acids research*, **34**, D354–7.

Kanehisa,Minoru *et al.* (2010) KEGG for representation and analysis of molecular networks involving diseases and drugs. *Nucleic Acids Res.*, **38**, D355–60.

Kitano,H. (2007) A robustness-based approach to systems-oriented drug design. *Nature reviews. Drug discovery*, **6**, 202–10.

Kitano,H. (2004) Biological robustness. *Nature reviews. Genetics*, **5**, 826–37.

Kritz,M.V. *et al.* (2010) Organising metabolic networks: Cycles in flux distributions. *Journal of theoretical biology*, **265**, 250–60.

Larhlimi,A. *et al.* (2011) Robustness of metabolic networks: a review of existing definitions. *Bio Systems*, **106**, 1–8.

Li,J. V *et al.* (2011) Metabolic surgery profoundly influences gut microbial-host metabolic cross-talk. *Gut*, **60**, 1214–23.

Li,Ying *et al.* (2011) Exogenous stimuli maintain intraepithelial lymphocytes via aryl hydrocarbon receptor activation. *Cell*, **147**, 629–40.

Locasale,J.W. and Cantley,L.C. (2010) Altered metabolism in cancer. *BMC biology*, **8**, 88.

Low,P.S. *et al.* (2008) Discovery and development of folic-acid-based receptor targeting for imaging and therapy of cancer and inflammatory diseases. *Accounts of chemical research*, **41**, 120–9.

Lu,W. *et al.* (2009) Analytical strategies for LC-MS-based targeted metabolomics. *J Chromatogr B Analyt Technol Biomed Life Sci.*, **871**, 236–42.

Ma,H.-W. *et al.* (2004) Decomposition of metabolic network into functional modules based on the global connectivity structure of reaction graph. *Bioinformatics (Oxford, England)*, **20**, 1870–6.

Maier,K. *et al.* (2010) Dynamics and control of the central carbon metabolism in hepatoma cells. *BMC systems biology*, **4**, 54.

Martin,F.J. *et al.* (2010) Dietary Modulation of Gut Functional Ecology Studied by Fecal Metabonomics. *J Proteome Res.*, **9**, 5284–5295.

Martinez-Rivas,J. and Vega,J. (1998) Purification and characterization of NAD-isocitrate dehydrogenase from chlamydomonas reinhardtii. *Plant physiology*, **118**, 249–55.

Ma'ayan,A. (2009) Insights into the organization of biochemical regulatory networks using graph theory analyses. *The Journal of biological chemistry*, **284**, 5451–5.

Montañez,R. *et al.* (2010) When metabolism meets topology: Reconciling metabolite and reaction networks. *BioEssays : news and reviews in molecular, cellular and developmental biology*, **32**, 246–56.

Nelson,D.L. and Cox,M.M. (2008) Lehninger principles of biochemistry Wh Freeman.

Newman,M. and Girvan,M. (2004) Finding and evaluating community structure in networks. *Physical Review E*, **69**, 1–15.

Newman,M.E.J. (2006) Modularity and community structure in networks. *Proceedings of the National Academy of Sciences of the United States of America*, **103**, 8577–82.

Nolan,R.P. *et al.* (2006) Identification of distributed metabolic objectives in the hypermetabolic liver by flux and energy balance analysis. *Metabolic engineering*, **8**, 30–45.

Nolan,R.P. and Lee,K. (2012) Dynamic model for CHO cell engineering. *Journal of biotechnology*, **158**, 24–33.

Nolan,R.P. and Lee,K. (2010) Dynamic model of CHO cellmetabolism. *Metabolic engineering*, **13**, 108–124.

Oda,K. *et al.* (2005) A comprehensive pathway map of epidermal growth factor receptor signaling. *Molecular systems biology*, **1**, 2005.0010.

Ogata,H. *et al.* (1999) KEGG: Kyoto Encyclopedia of Genes and Genomes. *Nucleic acids research*, **27**, 29–34.

Ortega,F. and Acerenza,L. (2011) Modular metabolic control analysis of large responses in branched systems – application to aspartate metabolism. **278**, 2565–2578.

Ozer,N. *et al.* (2001) Kinetic properties of human placental glucose-6-phosphate dehydrogenase. *The international journal of biochemistry & cell biology*, **33**, 221–6.

Papin,J. a *et al.* (2004) Hierarchical thinking in network biology: the unbiased modularization of biochemical networks. *Trends in biochemical sciences*, **29**, 641–7.

Parekh,S. and Anania,F. a (2007) Abnormal lipid and glucose metabolism in obesity: implications for nonalcoholic fatty liver disease. *Gastroenterology*, **132**, 2191–207.

Petersen,K.F. *et al.* (1995) Triiodothyronine treatment increases substrate cycling between pyruvate carboxylase and malic enzyme in perfused rat liver. *Metabolism*, **44**, 1380–3.

Potapov,A.P. *et al.* (2008) The pairwise disconnectivity index as a new metric for the topological analysis of regulatory networks. *BMC bioinformatics*, **9**, 227.

Pothenf,A. *et al.* (1990) Partitioning Sparse Matrices with Eigenvectors of Graphs. *Siam J Matrix Anal. Appl.*, **11**, 430–452.

Prill,R.J. *et al.* (2005) Dynamic properties of network motifs contribute to biological network organization. *PLoS biology*, **3**, e343.

Qin,S. *et al.* (2007) Pioglitazone stimulates apolipoprotein A-I production without affecting HDL removal in HepG2 cells: involvement of PPAR-alpha. *Arteriosclerosis, thrombosis, and vascular biology*, **27**, 2428–34.

Quinton-Tulloch,M.J. *et al.* (2013a) Trade-off of dynamic fragility but not of robustness in metabolic pathways in silico. *The FEBS journal*, **280**, 160–73.

Quinton-Tulloch,M.J. *et al.* (2013b) Trade-off of dynamic fragility but not of robustness in metabolic pathways in silico. *The FEBS journal*, **280**, 160–73.

Ravasz,E. *et al.* (2002) Hierarchical organization of modularity in metabolic networks. *Science (New York, N.Y.)*, **297**, 1551–5.

Reidy,S.P. and Weber,J.-M. (2002) Accelerated substrate cycling: a new energy-wasting role for leptin in vivo. *American journal of physiology. Endocrinology and metabolism*, **282**, E312–7.

Reznik,E. and Segre,D. (2010) On the stability of metabolic cycles. **266**, 536–549.

Riel,N.A.W. Van and Sontag,E.D. Parameter estimation in models combining signal transduction and metabolic pathways : the dependent input approach. *Engineering and Technology*, 263–274.

Rippa,M. *et al.* (1998) 6-Phosphogluconate dehydrogenase: the mechanism of action investigated by a comparison of the enzyme from different species. *Biochimica et biophysica acta*, **1429**, 83–92.

Saez-Rodriguez,J. *et al.* (2008) Automatic decomposition of kinetic models of signaling networks minimizing the retroactivity among modules. *Bioinformatics (Oxford, England)*, **24**, i213–9.

Saezrodriguez,J. *et al.* (2005) Dissecting the puzzle of life: modularization of signal transduction networks. *Computers & Chemical Engineering*, **29**, 619–629.

Sazanov,L. a and Jackson,J.B. (1994) Proton-translocating transhydrogenase and NAD- and NADP-linked isocitrate dehydrogenases operate in a substrate cycle which contributes to fine regulation of the tricarboxylic acid cycle activity in mitochondria. *FEBS letters*, **344**, 109–16.

Schomburg,I. *et al.* (2002) BRENDA, enzyme data and metabolic information. *Nucleic acids research*, **30**, 47–9.

Schuster,S *et al.* (2002) Exploring the pathway structure of metabolism: decomposition into subnetworks and application to Mycoplasma pneumoniae. *Bioinformatics (Oxford, England)*, **18**, 351–61.

Sellick,C.A. *et al.* (2010) Evaluation of extraction processes for intracellular metabolite profiling of mammalian cells: matching extraction approaches to cell type and metabolite targets. *Metabolomics*, **6**, 427–438.

Shearer,H.L. *et al.* (2004) Characterization of NADP-dependent malic enzyme from developing castor oil seed endosperm. *Archives of biochemistry and biophysics*, **429**, 134–44.

Si,Y. *et al.* (2007) Flux profile and modularity analysis of time-dependent metabolic changes of de novo adipocyte formation. *American journal of physiology. Endocrinology and metabolism*, **292**, E1637–46.

Si,Y. *et al.* (2009) Impact of perturbed pyruvate metabolism on adipocyte triglyceride accumulation. *Metabolic engineering*, **11**, 382–90.

Siddiquee,K.A.Z. *et al.* (2004) Metabolic flux analysis of pykF gene knockout Escherichia coli based on 13C-labeling experiments together with measurements of enzyme activities and intracellular metabolite concentrations. *Applied microbiology and biotechnology*, **63**, 407–417.

Smith,M.T. (2003) Mechanisms of troglitazone hepatotoxicity. *Chemical research in toxicology*, **16**, 679–87.

Sridharan,G.V. *et al.* (2011) Identification of Biochemical Network Modules Based on Shortest Retroactive Distances. *PLoS Computational Biology*, **7**.

Sridharan,G.V. *et al.* (2012) Metabolic flux-based modularity using shortest retroactive distances. *BMC systems biology*, **6**, 155.

Stelling,J. *et al.* (2004) Robustness of cellular functions. *Cell*, **118**, 675–85.

Steuer,R. *et al.* (2007) From structure to dynamics of metabolic pathways: application to the plant mitochondrial TCA cycle. *Bioinformatics (Oxford, England)*, **23**, 1378–85.

Steuer,R. *et al.* (2006) Structural kinetic modeling of metabolic networks. *Proceedings of the National Academy of Sciences of the United States of America*, **103**, 11868–73.

Tang,X. *et al.* (2011) A comparison of the functional modules identified from time course and static PPI network data. *BMC bioinformatics*, **12**, 339.

Taylor,I.W. *et al.* (2009) Dynamic modularity in protein interaction networks predicts breast cancer outcome. *Nature biotechnology*, **27**, 199–204.

Terzer,M. and Stelling,J. (2008) Large-scale computation of elementary flux modes with bit pattern trees. *Bioinformatics (Oxford, England)*, **24**, 2229–35.

TH,C. (2009) Introduction to Algorithms MIT Press., Cambridge, Massachusetts.

Tisdale,Michael J (2005) Molecular pathways leading to cancer cachexia. *Physiology (Bethesda, Md.)*, **20**, 340–8.

Treviño,S. *et al.* (2012) Robust Detection of Hierarchical Communities from Escherichia coli Gene Expression Data. *PLoS Computational Biology*, **8**, e1002391.

Tseng,Y.-H. *et al.* (2010) Cellular bioenergetics as a target for obesity therapy. *Nature reviews. Drug discovery*, **9**, 465–82.

Turnbaugh,P.J. *et al.* (2006) An obesity-associated gut microbiome with increased capacity for energy harvest. *Nature*, **444**, 1027–31.

Turnbaugh,P.J. and Gordon,J.I. (2009) The core gut microbiome, energy balance and obesity. *J Physiol.*, **587**, 4153–8.

Vaahtovuo,J. *et al.* (2001) Study of murine faecal microflora by cellular fatty acid analysis; effect of age and mouse strain. *Antonie van Leeuwenhoek*, **80**, 35–42.

Variano,E. a. and Lipson,H. (2004) Networks, Dynamics, and Modularity. *Physical Review Letters*, **92**, 188701.

Varma,A. and Palsson,B.O. (1994) Stoichiometric flux balance models quantitatively predict growth and metabolic by-product secretion in wild-type Escherichia coli W3110. *Applied and environmental microbiology*, **60**, 3724–3731.

Visser,D. (2003) Dynamic simulation and metabolic re-design of a branched pathway using linlog kinetics. *Metabolic Engineering*, **5**, 164–176.

Wang,Z. *et al.* (2011) Gut flora metabolism of phosphatidylcholine promotes cardiovascular disease. *Nature*, **472**, 57–63.

Want,E.J. *et al.* (2005) The expanding role of mass spectrometry in metabolite profiling and characterization. *Chembiochem*, **6**, 1941–51.

Whitehead,K. a *et al.* (2009) Knocking down barriers: advances in siRNA delivery. *Nature reviews. Drug discovery*, **8**, 129–38.

Whitt,D.D. and Demoss,R.D. (1975) Effect of Microflora on the Free Amino Acid Distribution in Various Regions of the Mouse Gastrointestinal Tract. *Appl Microbiol.*, **30**, 609.

Van de Wiele,T. *et al.* (2005) Human Colon Microbiota Transform Polycyclic Aromatic Hydrocarbons to Estrogenic Metabolites. *Environ Health Perspect*, **113**, 6–10.

Wikoff,W.R. *et al.* (2009) Metabolomics analysis reveals large effects of gut microflora on mammalian blood metabolites. *Proc Natl Acad Sci USA*, **106**, 3698–703.

Wilcox,D.M. *et al.* (2006) Delivery of RNAi reagents in murine models of obesity and diabetes. *Journal of RNAi and gene silencing : an international journal of RNA and gene targeting research*, **3**, 225–36.

Yook,S.-H. *et al.* (2004) Functional and topological characterization of protein interaction networks. *Proteomics*, **4**, 928–42.

Yoon,J. *et al.* (2007) Modular decomposition of metabolic reaction networks based on flux analysis and pathway projection. *Bioinformatics (Oxford, England)*, **23**, 2433–40.

Yousofshahi,M. *et al.* (2011) Probabilistic pathway construction. *Metab Eng.*, **13**, 435–44.

Yun,C.-H. *et al.* (2008) The T790M mutation in EGFR kinase causes drug resistance by increasing the affinity for ATP. *Proceedings of the National Academy of Sciences*, **105**, 2070–2075.

Zhao,J. *et al.* (2006) Hierarchical modularity of nested bow-ties in metabolic networks. *BMC bioinformatics*, **7**, 386.

Zheng,X. *et al.* (2011) The footprints of gut microbial-mammalian co-metabolism. *J Proteome Res.*, **10**, 5512–22.

Zhu,L. *et al.* (2012) Engineering the robustness of industrial microbes through synthetic biology. *Trends in microbiology*, **20**, 94–101.