EVOLUTION AND STOCHASTIC DYNAMICS IN COMPLEX SYSTEMS

by

ADITYA BALLAL

A dissertation submitted to the

School of Graduate Studies

Rutgers, The State University of New Jersey

In partial fulfillment of the requirements

For the degree of

Doctor of Philosophy

Graduate Program in Physics

Written under the direction of

Alexandre V. Morozov

and approved by

New Brunswick, New Jersey

January, 2022

ABSTRACT OF THE DISSERTATION

EVOLUTION AND STOCHASTIC DYNAMICS IN COMPLEX SYSTEMS

By ADITYA BALLAL

Dissertation Director:

Alexandre V. Morozov

The dissertation provides an insight on the mathematical modeling of two complex systems. Chapter 1-4 are about the biophysical model we developed for enzyme evolution. The model predicts the reaction rates as a function of the enzyme sequence. The free energies in Michaelis-Menten kinetics are represented as a function of one amino-acid contribution and two amino-acid pairs of the enzyme sequence. Our model predicts reaction-rates with high accuracy and relatively few coupling terms. The sparseness in coupling terms results in highly interpretable Michaelis-Menten energy landscapes which exhibit little non-linearity. Chapter 5-9 talk about modeling first-passage properties of random-walks on a community-structured network. The properties are exploited in developing an algorithm to partition the network into communities. On testing our algorithm on multiple artificial and real-world networks, we find that it performs better or as well as the other competitive algorithms.

Acknowledgements

I am deeply grateful to my PhD. advisor Alexandre Morozov for encouraging me to explore many topics. I learnt a lot under his guidance and the leads he provided when I got stuck were extremely helpful. I really enjoyed the independence I got while working under him. I am also grateful my collaborator Paul O'Maille; I gained a lot of knowledge in our weekly discussions on the enzyme evolution project. My gratitude also extends to Anirvan Sengupta; his comments during the group meetings provided good leads on the Community Detection project.

Next, I would also like to thank all my colleagues Willow Kion-Crosby, Unab Javed, Ted Malliaris and Jesus Rives for helping me with my work and some interesting random discussions. Weekly calls with Willow for the Community Detection project helped us shape the project.

I was fortunate to develop friendships with some really amazing people: Deepti Jain, Surya Teja Gavva, Amartya Singh, Jay Vora, Ruturaj Apte and Aditya Pottukuchi with whom I lived the better part of the last seven years. Sharing many activities with them like trying new food, having chai, hiking, traveling, cooking, playing tennis, watching bad movies made life fun. I would like to extend my gratitude to Gleb Koustov for some interesting discussions during our tea time break at Serin, Ghanshyam Khanal for some super imaginative cricket related discussions, Abhishek Bhrushundi for helping me with Community Detection project and creative names of theorems and lemmas, Abhijith Gandrakota for random discussions, Chinmay Khandekar and Abhinav Narain for some amazing hiking trips and Amey Bhangale for ping pong. Other than the names mentioned, I am also grateful to Jimmy Kim, Rahul Sharma and Kavish Seth for their support in these seven years.

I was lucky to be a part of the band "The Piano Axioms" with Karuna Sangam, Carlie Mullin, Danielle Slavis and Kendi Bailey. Social distance jamming with them helped me get through the pandemic. I would also like to thank the university choir for giving me opportunity to sing (and throat sing) and Rutgers salsa club for teaching how to dance. I am extremely thankful to David Cheifer for teaching me fundamentals of jazz improvisations. A special mention to the maestro Chick Corea whose amazing music always generates adrenaline in my body. I would like to thank for his autograph and his words of encouragement ”You sound good, keep playing”.

Lastly, I would like to thank my parents, my grandmother, my brother Chaitanya and our dog Bhalu for their love, constant support and encouragement without which completing my PhD. would have been impossible.

The first four chapters of this dissertation are original work from refs. [1] and [2].

Table of Contents

Chapter 1

Introduction to Enzyme Evolution

Understanding evolution is critical for understanding biology. For millions of years plants and animals have adapted themselves to suit the environment around them. Many of these adaptations involved production of highly specific compounds through mutation and selection of enzymes.

Our study involves the enzymes that catalyze the production of a type of complex hydrocarbons known as terpenes. This family of enzymes called terpene synthases (TPSs) is a major family of enzymes found in a variety of plants and insects ([3],[4]). Terpenes play a significant role for many organisms; they are involved in pollination, plant and insect predator defense mechanisms, and symbiotic relations. They are also widely used as flavors, fragrances and medicines; a well-known example of the latter is artemisinin, a naturally occurring anti-malarial drug extracted from Artemisia annua. Terpenes and terpenoids are the primary constituents of many essential oils in medicinal plants and flowers; examples include α-bisabolol, a monocyclic sesquiterpene alcohol which forms the basis of a colorless viscous oil from German chamomile (Matricaria recutita) and Myoporum crassifolium, and zingiberene, a monocyclic sesquiterpene that is the predominant constituent of ginger oil.

Enzymes in the TPS family are capable of converting several universal substrates into a diverse variety of terpene products. Amorpha-4,11-diene synthase (ADS) catalyzes the reaction to the production of amorpha-4,11-diene, the bicyclic hydrocarbon precursor of artemisinin, from farnesyl pyrophosphate (FPP), a linear substrate in the plant Artemisia annua. In the same plant, (E)-β-farnesene, a linear hydrocarbon is also produced from the same substrate FPP, the only difference being that it is catalyzed by a different enzyme (E)-β-farnesene synthase (BFS).

Thus, in order to analyze the molecular mechanisms of emergence of terpene cyclization, our experimental collaborators led by Dr. Paul O'Maille carried out in-depth examination of mutational space around (E)-β-farnesene synthase using structure-based combinatorial protein engineering (SCOPE). They employed multiple rounds of mutation and selection to map out the ensemble of mutational pathways and identify sequence variants with novel enzymatic functions. Functional assays, carried out for more than a hundred sequence variants, involve enzyme characterization by mass-spectroscopy and kinetic measurements.

To characterize the biophysical landscape of our mutant libraries, we have developed a model to predict the reaction rates as a function of the enzyme sequence. The model represents free energies in Michaelis-Menten kinetics as a function of one amino-acid contribution and two amino-acid pairs of the enzyme sequence. Our model predicts reaction-rates with high accuracy and relatively few coupling terms. The sparseness in coupling terms results in highly interpretable Michaelis-Menten energy landscapes which exhibit little epistasis.

We have used our spin-glass-like models of Michaelis-Menten landscapes to develop a hierarchy of biophysical models of enzyme fitness, interpreting the latter in terms of the protein's ability to catalyze reactions beneficial to the cell while minimizing production of deleterious or unwanted by-products. We have found that, compared to the free energy landscapes, biophysical fitness landscapes are more epistatic.

The experiments performed by Dr. Paul O'Maille and his team are mentioned in chapter 2. Chapter 3 talks about the biophysical model and models for fitness of an enzyme.

Chapter 2

Experiments

In order to investigate TPS evolution in *A. annua* systematically, Dr. O'Maille and his team used structural analysis to identify 24 variable positions within 6 Å of the (E)-β-farnesene synthase (BFS) active site center that differed between BFS and amorpha-4,11-diene synthase (ADS). Using structure-based combinatorial protein engineering (SCOPE) ([5]) they introduced ADS mutations into the BFS sequence at the 24 positions, thus constructing a mutant library. After the initial screening step where enzymes in the library were tested for solubility and biochemical activity, it was found that the ADS mutation at position 474 produced an inactive enzyme (all residue numbers are relative to the *A. annua* BFS sequence). In addition, A395G and G431A mutations which do not correspond to the ADS sequence were inadvertently introduced into the library, resulting in biochemically active enzymes. In-depth characterization of $\sim$100 variants of biochemically active enzymes which had one or more mutations at the 25 positions (including 395 and 431 but excluding 474) were carried out. Although a significant amorpha-4,11-diene production was not observed in any of the mutants, Dr. O'Maille and his team identified several TPSs which produce sizable quantities of α-bisabolol – a major product of TPS enzymes in *A. annua* and *Asteracea* plants. This made them focus on a highly specific α-bisabolol-producing mutant which contains 5 mutations with respect to the *A. annua* BFS.

To identify which residues were responsible for the α-bisabolol activity, Dr. O'Maille and his team designed a library, M5, that consisted of all combinations of the 5 amino acid mutations ($2^5 = 32$ sequences in total) bridging BFS and previously discovered α-bisabolol -producing BFS variant (Fig. 2.1,2.2). Using SCOPE, the M5 library was synthesized and the clones were verified by sequencing. Next, all recombinant enzymes

.

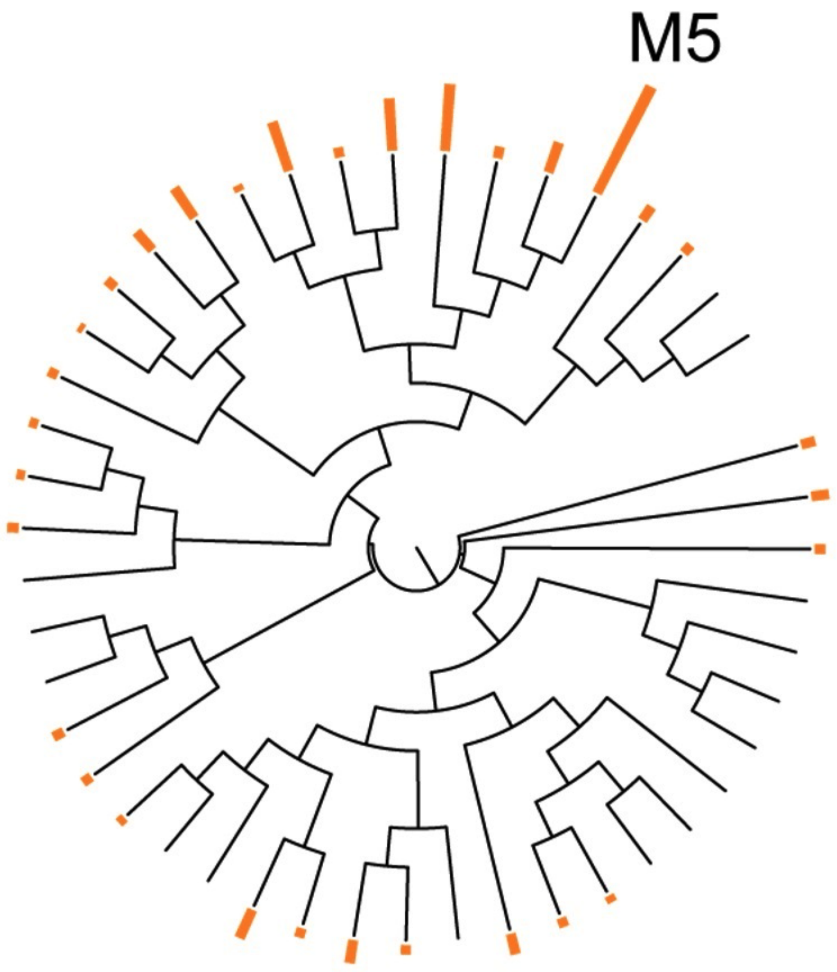


Figure 2.1: A phylogenetic tree of BFS variants from Salmon et al. ([6]) was created using ClustalW ([7]). The resulting tree was annotated using the Interactive Tree of Life ([8]) according to the percentage of α-bisabolol products produced (orange bars). The M5 mutant is labeled.

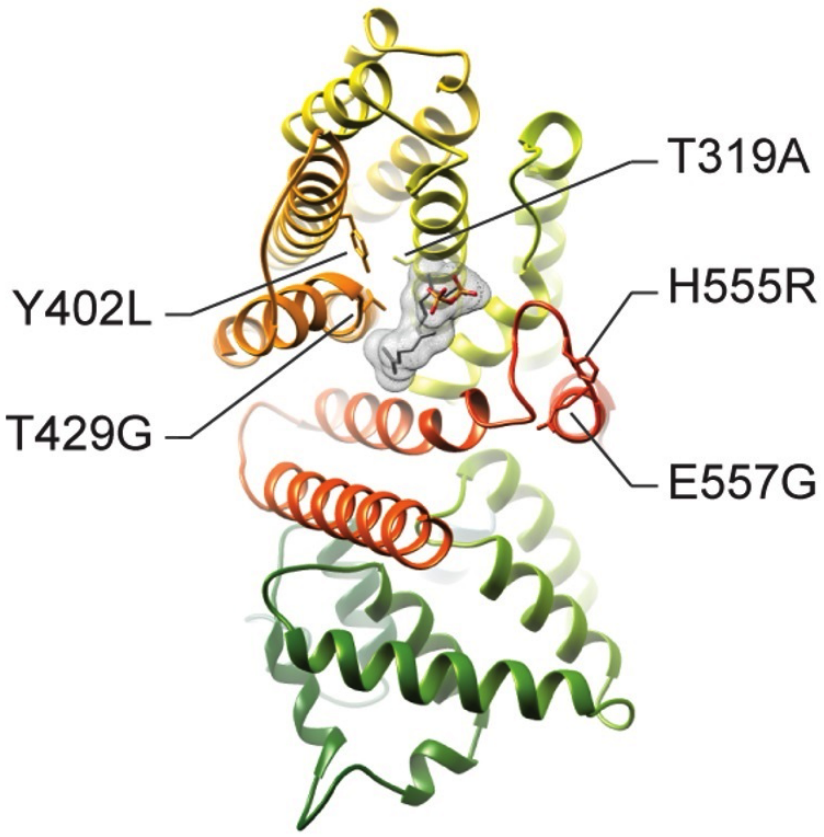


Figure 2.2: Structural positions of residue substitutions in the M5 mutant used for the M5 library synthesis and characterization

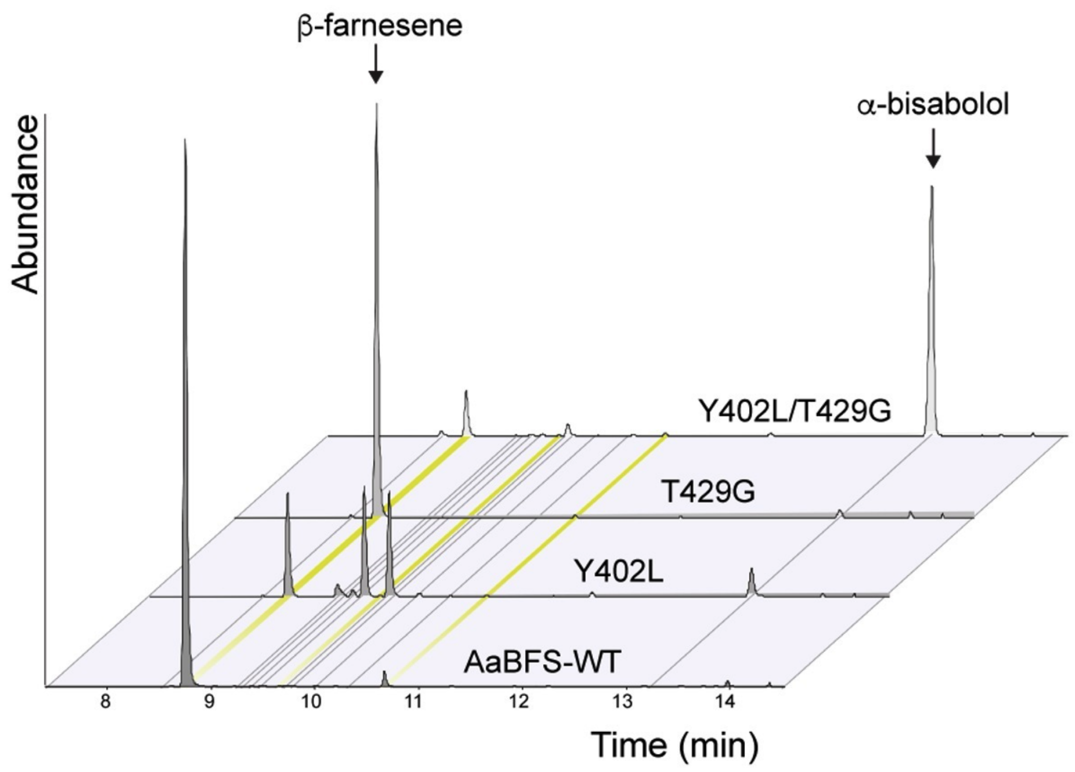


Figure 2.3: GC chromatograms of select members of characterized mutants, with major products indicated

were characterized for product specificity by GC-MS ([9], [10]) and their kinetic properties were measured using MGA ([11]). Consistent with the observations based on the *A. annua* BFS 6 Å library discussed above ([6]), the Y402L substitution was essential for product cyclization: in the absence of Y402L, all mutants produced linear terpene products. Of the cyclic-producing variants, two product profiles were evident, either a multiple product profile (as seen with the Y402L single mutant) or α-bisabolol as the dominant product in the profile (Fig. 2.3). It was observed that α-bisabolol product specificity was primarily attributable to a single additional mutation T429G in the Y402L background (Fig. 2.3), whereas the presence of additional mutations (T319A, H555R and E557G) had weaker effects on product specificity. However, kinetic analysis revealed that the total kinetic rate k_{cat} was affected significantly by the additional mutations in the T429G/Y402L background. In particular, the C-terminal H555R mutation was very detrimental to catalytic activity. Alone, H555R resulted in a 66% reduction in enzyme activity compared to the BFS wild-type (BFS-WT) enzyme; and in combination with the other mutations, enzyme activity was further reduced to between 1 and 6% of BFS-WT activity. In comparison, the α-bisabolol -producing Y402L/T429G mutant (69% of the total output is α-bisabolol) has moderate catalytic activity (26% of BFS-WT activity) comparable to other native and specific TPS enzymes.

Chapter 3

Biophysical Model

3.1 Quantitative description of enzyme libraries

We consider two libraries of mutant enzymes. Each enzyme in the library can have mutations at up to L variable positions compared to the wild-type sequence, with the amino acid alphabet at each position allowed to be in one of the two states: wild type (W) or mutant (M). Thus, each enzyme sequence S_j can be represented by

$$S_j = A_1^{(j)} A_2^{(j)} \dots A_L^{(j)}, \qquad j = 1, \dots, N, \tag{3.1}$$

where $A_k^{(j)} = [W, M]$ is the amino acid at position k in sequence j and N is the total number of sequences in a given library. Note that k numbers variable positions rather than absolute amino acid positions within a sequence, and that the rest of the sequence outside of the L positions is invariant. For each enzyme in both libraries, reaction rates for $n = 11$ distinct products ($k_{cat,i}, i = 1, \dots, n$) have been inferred using GC-MS and MGA (several other products had negligibly low rates and are therefore excluded from this study). The first library, the *A. annua* BFS 6 Å library which we shall refer to as M25, contains $k_{cat,i}$ values for 93 distinct sequences, including the wild-type, with mutations at up to 25 variable positions ([6]). The second library, which was described above as the M5 library, contains $2^5 = 32$ sequences, including the wild-type, for all possible combinations of mutations at 5 positions (319, 402, 429, 555, 557) which separate BFS from the novel α-bisabolol producing enzyme originally found in the M25 library. The combined library contains $N = 122$ distinct sequences, including BFS-WT, with mutations at one or more among 25 variable positions.

3.2 Enzyme kinetics

We have modeled enzymatic reaction rates using the Michaelis-Menten model of enzyme kinetics ([12], [13]). According to this model, enzymes catalyze chemical reactions in a two-step process. The first step is a reversible reaction where a substrate molecule binds the enzyme's active site. In the second reaction, assumed to be irreversible, substrate is transformed into product and released from the enzyme. In general, terpene synthases in our libraries catalyze multiple reactions simultaneously starting from the same substrate. We assume that the first step is the same in all these reactions since it involves just the substrate and the enzyme:

$$\begin{tikzcd} & & \mathrm{P_1 + E} \\ & & \mathrm{P_2 + E} \\ \mathrm{S + E} \arrow[r, shift left, "k_1"] & \mathrm{E \cdot S} \arrow[l, shift left, "k_{-1}"] \arrow[uur, "k_{cat,1}"] \arrow[ur, "k_{cat,2}"] \arrow[r, "k_{cat,3}"] \arrow[ddr, "k_{cat,n}"] & \mathrm{P_3 + E} \\ & & \vdots \\ & & \mathrm{P_n + E} \end{tikzcd}$$

Here, S is the substrate, E is the enzyme, $\mathrm{P_i}$'s are the products, and $k_{cat,i}$ denotes the reaction rate for product i. Each reaction rate $k_{cat,i}$ of an enzyme with sequence S_j depends on the Gibbs free energies G_3 and $G_{4,i}$ (Fig. 3.1):

$$k_{cat,i}\left(S_j\right) = B_i \exp\left(-\frac{G_{4,i}\left(S_j\right) - G_3\left(S_j\right)}{k_B T}\right), \tag{3.2}$$

where B_i is the reaction rate for product i in the absence of the free energy barrier, k_B is the Boltzmann constant, and T is the temperature. As discussed above, we assume that G_3 is independent of the product for a given enzyme, while $G_{4,i}$ is product-specific. Note that the total reaction rate is given by $k_{cat}\left(S_j\right) = \sum_i k_{cat,i}\left(S_j\right)$ and that the probability $c_i$of producing product i is therefore $c_i\left(S_j\right) = k_{cat,i}\left(S_j\right)/k_{cat}\left(S_j\right)$. Thus, the observed reaction rates are given by $k^{obs}_{cat,i}\left(S_j\right) = c_i\left(S_j\right) k_{cat}\left(S_j\right)$, where $c_i\left(S_j\right)$ are relative abundances inferred from GC-MS data and $k_{cat}\left(S_j\right)$ are measured using MGA. In the MGA assay, the $k_{cat}\left(S_j\right)$ values were predicted using a linear fit to enzyme velocities at a fixed substrate concentration and a series of enzyme concentrations. These experiments were carried out in triplicate but due to non-linearities and/or noise in the

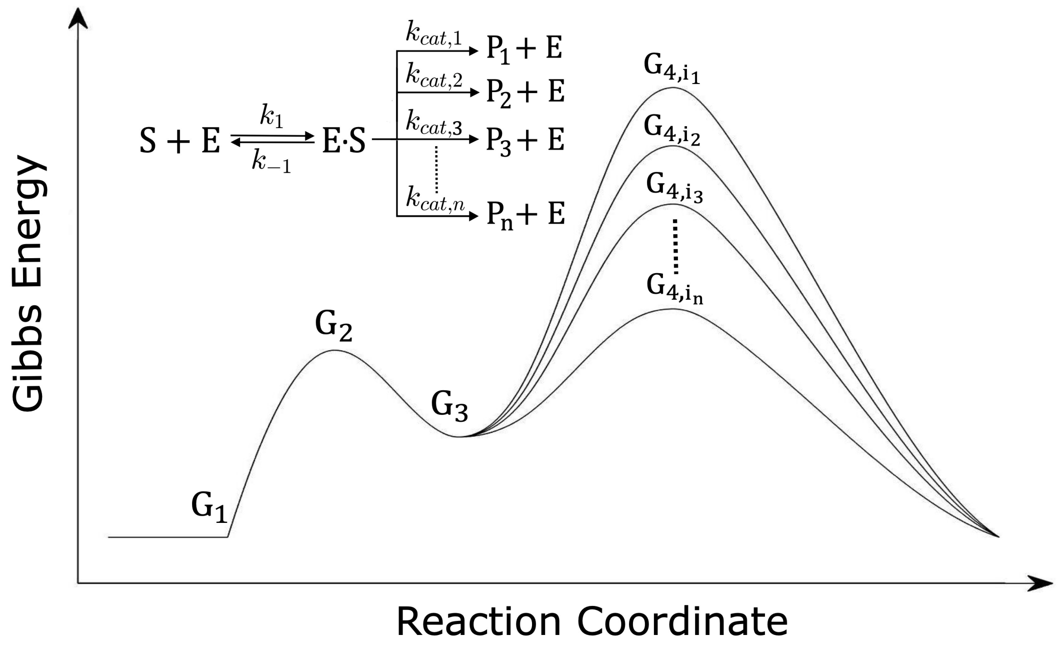


Figure 3.1: **Michaelis-Menten model of enzyme kinetics:** Shown are free energy profiles for converting substrate S into products $P_1 \dots P_n$, catalyzed by the enzyme E. G_1, G_2, G_3 and G_{4,i_k} are free energies at the various stages of the enzymatic reactions, and k_{-1}, k_1, k_{cat,i_k} are the corresponding reaction rates as shown in the inset (product indices $i_1 \dots i_n$ are sorted in the decreasing order of G_{4,i_k} in the panel). Each reaction rate k_{cat,i_k} depends on the difference between free energies $G_{4,i}$ and G_3 (Eq. (2)). Inset shows the corresponding kinetic rates of the Michaelis-Menten reaction.

data not all measurement series could be reliably fit to extract the k_{cat} values, resulting in 1-3 independent $k_{cat}(S_j)$ measurements that were subsequently averaged to compute $k_{cat,i}^{obs}(S_j)$. Likewise, the GC-MS experiments were carried out in triplicate, with the relative abundances $c_i(S_j)$ averaged prior to being employed in the computation of product-specific kinetic rates.

We find that the enzymes in the combined library are characterized by a wide range of kinetic rates depending on the product type (Fig. 3.2).

3.3 Pairwise model of enzyme energetics

To model enzyme kinetics and energetics, we have employed a pairwise model inspired by spin-glass models in statistical physics ([14]; [15]). Such models have been extensively used to study protein stability and protein-protein interactions ([16], [17], [18], [19]).

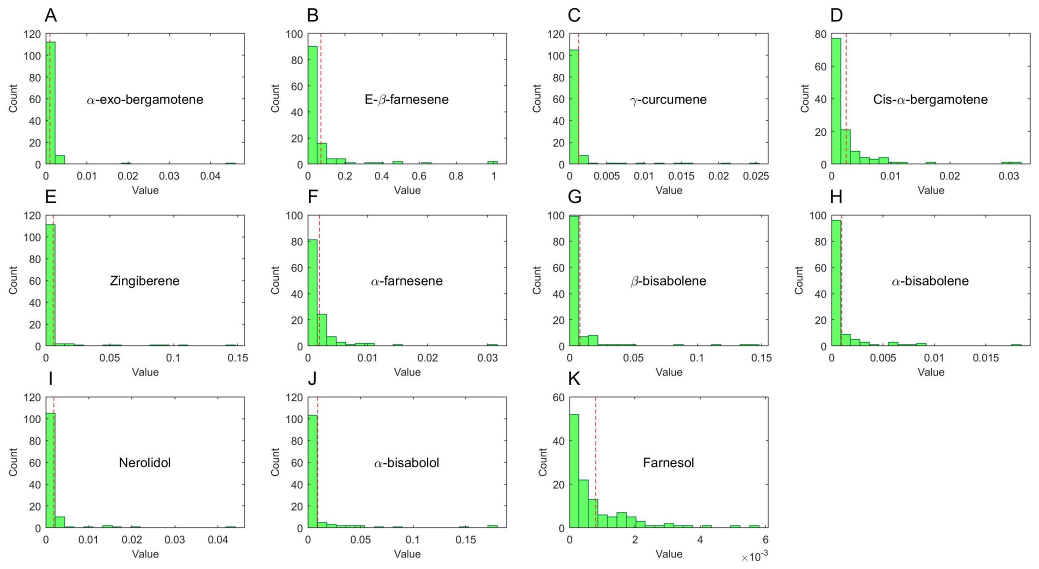


Figure 3.2: Each panel shows a 20-bin histogram of and $\tilde{k}^{obs}_{cat,i}(S_j)$values for each product i. Vertical dashed lines indicate mean values of each $\tilde{k}^{obs}_{cat,i}$ distribution

Unlike these previous approaches, which typically use protein sequence alignments as input to generating novel sequences and scoring the existing ones, our model is designed to predict reaction rates $k_{cat,i}$ as a function of the enzyme's sequence. For a given enzyme, we represent the Gibbs free energies G_3 and $G_{4,i}$ for each product i as a sum over single-amino-acid (aa) terms and two-amino-acid coupling terms:

$$G_3\left(S_j\right) = \sum_{k=1}^{L} H_k\left(A_k^{(j)}\right) + \sum_{(k,l)} E_{kl}\left(A_k^{(j)}, A_l^{(j)}\right), \tag{3.3}$$

$$G_{4,i}\left(S_j\right) = \sum_{k=1}^{L} h_k^{(i)}\left(A_k^{(j)}\right) + \sum_{(k,l)} \varepsilon_{kl}^{(i)}\left(A_k^{(j)}, A_l^{(j)}\right), \tag{3.4}$$

where $H_k/h_k^{(i)}$and $E_{kl}/\varepsilon_{kl}^{(i)}$ are the one-aa and two-aa contributions to G_3 and $G_{4,i}$, respectively, and the (k,l) sum in the second term on the right-hand side runs over all pairs of variable amino acids in each sequence. This is a set of $N\left(n+1\right) = 1464$ equations for the combined dataset. To reduce the number of parameters, we set all terms containing one or two wild-type amino acids to zero: $H_k\left(W\right) = h_k^{(i)}\left(W\right) = 0$, $E_{kl}\left(W,W\right) = E_{kl}\left(W,M\right) = E_{kl}\left(M,W\right) = 0$, $\varepsilon_{kl}^{(i)}\left(W,W\right) = \varepsilon_{kl}^{(i)}\left(W,M\right) = \varepsilon_{kl}^{(i)}\left(M,W\right) = 0$ ([18]). This guarantees that all G_3 and $G_{4,i}$ values are zero automatically for wild-type sequences, while leaving enough degrees of freedom to model one-aa effects (through the $H_k(M)/h_k^{(i)}(M)$ terms) and two-aa couplings (through the $E_{kl}(M,M)/\varepsilon_{kl}^{(i)}(M,M)$ terms). For the combined library and for all Gibbs free energies G_3 and $G_{4,i}$, this procedure yields up to $(n+1) = 12$ non-zero one-aa terms at each of $L = 25$variable positions and, similarly, up to 12 non-zero two-aa coupling terms at each of $L_p = 138$ pairs of variable positions. Note that, by construction, only two-aa couplings for pairs of positions at which all four aa combinations are available: (W,W), (W,M), (M,W) and (M,M) are included into the model.

3.3.1 Model Selection

Using Eqs. (3.2), (3.3) and (3.4), we obtain a set of $Nn = 1342$equations for predicting relative reaction rates:

$$\tilde{k}_{cat,i}\left(S_j\right) = exp\left\{-\frac{1}{k_BT}\left(\sum_k h_k^{(i)}\left(A_k^{(j)}\right) + \sum_{(k,l)}\varepsilon_{kl}^{(i)}\left(A_k^{(j)},A_l^{(j)}\right)\right.\right.$$
$$\left.\left.-\sum_k H_k\left(A_k^{(j)}\right) - \sum_{(k,l)} E_{kl}\left(A_k^{(j)},A_l^{(j)}\right) - C_i\right)\right\}, \tag{3.5}$$

where $\tilde{k}_{cat,i}\left(S_j\right) = k_{cat,i}\left(S_j\right)/k_{cat}^{obs}\left(WT\right)$ is the predicted relative reaction rate for product i ($k_{cat}^{obs}\left(WT\right)$ is the observed total reaction rate of the wild-type sequence), and the sequence-independent offsets C_i are defined by $e^{C_i/k_BT} = B_i/k_{cat}^{obs}\left(WT\right)$ (B_i is the reaction rate for product i of the wild-type sequence, Eq. (3.2)). This set of equations has $(n+1)\left(L+L_p\right)+n = 1967$ fitting parameters since there are L one-body terms, L_p coupling terms for each $G_{4,i}$ and G_3, and one additional term per product for the sequence-independent offset C_i. The predicted relative reaction rates are fitted to $\tilde{k}_{cat,i}^{obs}\left(S_j\right) = k_{cat,i}^{obs}\left(S_j\right)/k_{cat}^{obs}\left(WT\right)$, the relative reaction rates observed for each enzyme in the combined library.

Since the total number of fitting variables is greater than the number of measurements, we employ the LASSO (Least Absolute Shrinkage and Selection Operator) algorithm ([20], [21]), which reduces the number of non-zero fitting parameters by imposing a penalty proportional to their L^1 norm. We impose different penalties on one-body terms and two-body couplings, while the C_i offsets are left unconstrained. The problem therefore reduces to finding a set of fitting parameters which minimize the following expression:

$$\begin{aligned} \min_{h_k^{(i)}, H_k, \varepsilon_{kl}^{(i)}, E_{kl}, C_i} \Bigg\{ & \frac{1}{Nn} \sum_{i=1}^{n} \sum_{j=1}^{N} \left| \tilde{k}_{cat,i}\left(S_j\right) - \tilde{k}_{cat,i}^{obs}\left(S_j\right) \right|^2 \\ & + \lambda \left(\sum_{i=1}^{n} \sum_{k=1}^{L} \left| h_k^{(i)} \right| + \sum_{k=1}^{L} \left| H_k \right| \right) \\ & + \beta\lambda \left(\sum_{i=1}^{n} \sum_{(k,l)} \left| \varepsilon_{kl}^{(i)} \right| + \sum_{(k,l)} \left| E_{kl} \right| \right) \Bigg\}, \end{aligned} \tag{3.6}$$

where λ and $\beta\lambda$ are the regularization coefficients which determine the relative importance of the L^1 penalty terms. We have determined the penalty parameters λ and β by 4-fold cross validation. All enzyme sequences S_j were randomly partitioned into 4 equal-sized samples. One sample was assigned as the test set and the other 3 as the training set on which the model was fitted. This procedure was repeated 4 times, with each sample used exactly once as the test set. We varied $\lambda$from $10^{-2.8}$ to $10^{-1.4}$ and β from 1 to 10. For each pair of λ and β, we calculated the mean-square error in predicting the test set (the first term in Eq. (3.6)) and averaged it over all 4 cross-validation runs (Fig. 3.3). The error was minimized for $\lambda = 0.0079$ and $\beta = 2.7384$; these values were subsequently used to fit the model on the entire data set.

3.3.2 Checking validity of the model by synthetic data

Since the total number of potentially non-zero fitting parameters is larger than the number of reaction rate measurements in the combined library, we have additionally checked the consistency of the LASSO procedure by fitting our model to artificially generated data. To generate the artificial data, we randomly chose 7 non-zero one-body terms and 4 non-zero couplings for each $G_{4,i}$ and G_3 landscape and for each enzyme sequence (all other terms were assumed to be zero). These parameters were assigned random values based on two Gaussian distributions which were obtained by computing the mean m and the standard deviation σ of the one-body terms ($m = 0.24, \sigma = 0.85$)

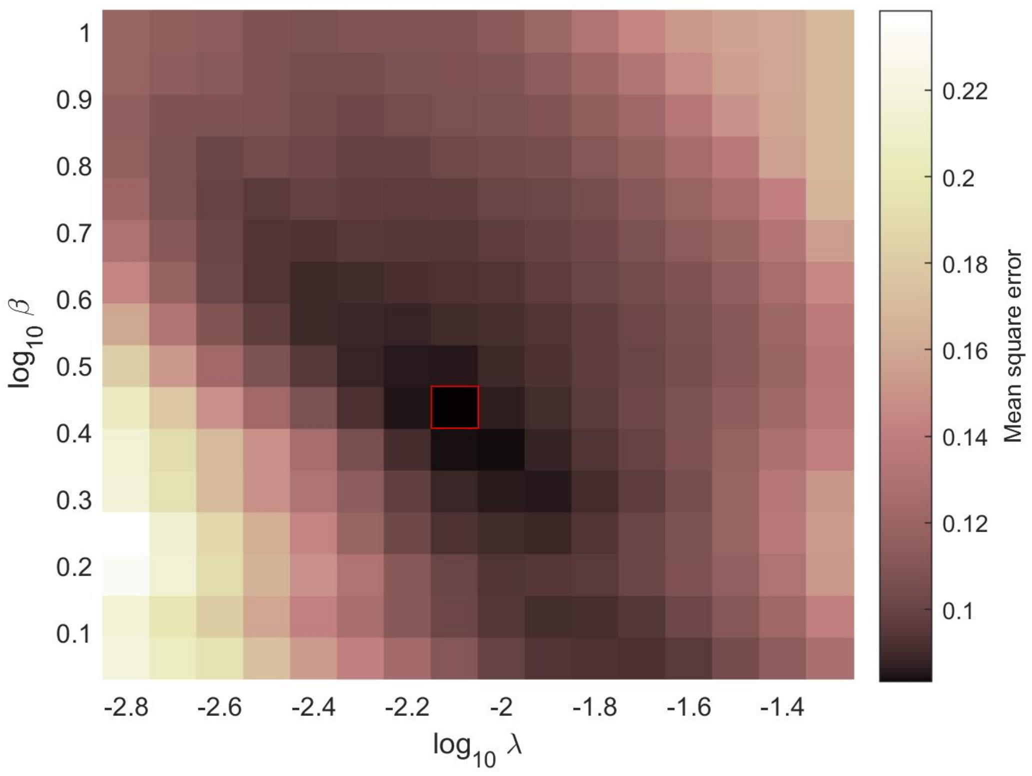


Figure 3.3: **Mean-square error as a function of the LASSO penalty parameters λ and β:** Shown are mean-square errors on the test set obtained via four-fold cross validation. The square with the minimum mean-square error, highlighted in red, was used to choose the optimal LASSO parameters.

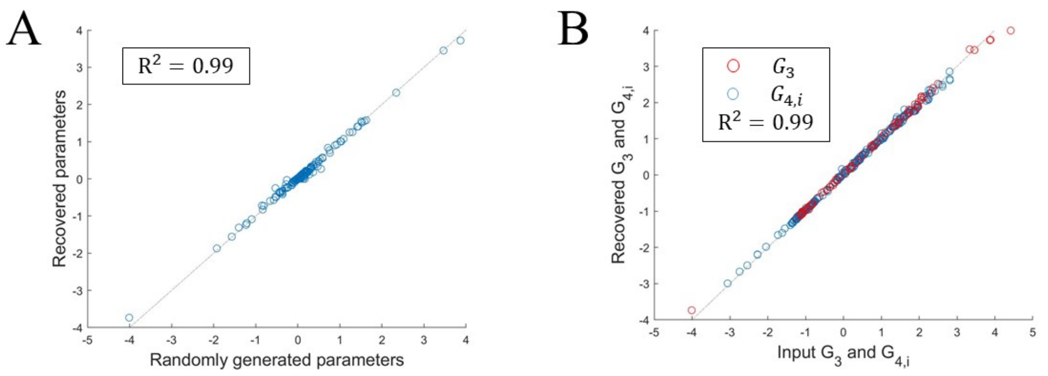


Figure 3.4: **Pairwise model predictions on synthetic data:** A set of artificial reaction rates $k'_{cat,i}(S_j)$ was generated using a pairwise model with randomly sampled parameters. A LASSO fit with cross-validation analogous to that used with real data was employed to recover the parameters of the model. Shown are the comparison between synthetically generated and predicted one- and two-aa model parameters (A) and the corresponding free energy values (B).

and the couplings ($m = -0.09, \sigma = 0.63$) inferred from the combined library. The sequence-independent offsets C_i were likewise sampled from a Gaussian distribution with $m = -5.52$ and $\sigma = 1.96$, where the Gaussian was fit to the set of C_i's obtained after fitting the model to the reaction rate data in the combined library. We have generated artificial $\tilde{k}'_{cat,i}$ values using these randomly chosen parameters:

$$\begin{aligned}\tilde{k}'_{cat,i}\left(S_j\right) = \exp\Bigg\{-\frac{1}{k_BT}\Bigg(&\sum_k h_k^{(i)}\left(A_k^{(j)}\right) + \sum_{(k,l)} \varepsilon_{kl}^{(i)}\left(A_k^{(j)}, A_l^{(j)}\right) \\ &-\sum_k H_k\left(A_k^{(j)}\right) - \sum_{(k,l)} E_{kl}\left(A_k^{(j)}, A_l^{(j)}\right) - C_i\Bigg)\Bigg\} + r,\end{aligned} \tag{3.7}$$

where r was randomly sampled from a Gaussian distribution with $m_\epsilon = 0.0$ and $\sigma_\epsilon = 0.003$ (σ_ϵ is the mean-square error obtained after fitting the model to the reaction rate data in the combined dataset). This artificial data set was treated as reaction rate measurements, and the parameters of the model as well as free energy values were subsequently inferred using LASSO as described above with a high accuracy(Fig. 3.4).

3.3.3 Results

We find that our fitting procedure yields sparse solutions for Michaelis-Menten free energy landscapes. Indeed, our collection of models for G_3 and $G_{4,i}$ fitted to the combined

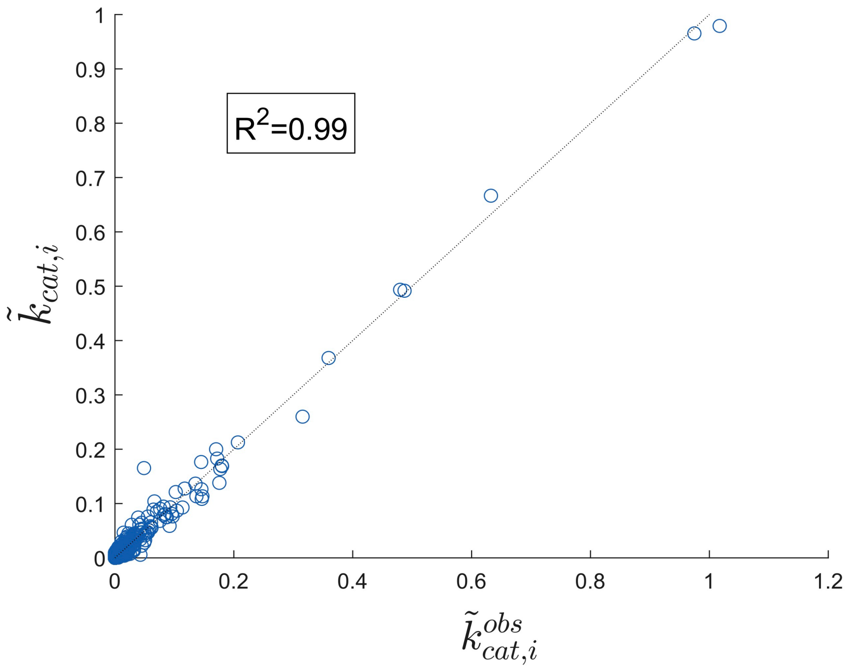


Figure 3.5: Michaelis-Menten reaction rates predicted using the pairwise model with cross-validation

data set with optimal one-body and two-body penalties contains a total of 113 out of $(n+1)\,L$=300 possible one-body terms and just 54 out of $(n+1)L_p = 1656$ possible two-body couplings, that is, on average, 9.4 out of 25 possible one-body terms and 4.5 out of 138 possible two-body couplings per free energy landscape. Thus, the observed $k_{cat,i}$ values of 11 products for 122 different sequences (1342 $k_{cat,i}$ values in total) are described using just 178 parameters: 113 one-body terms, 54 two-body couplings, and 11 sequence-independent offsets C_i. The model fits the reaction rate data with $R^2 = 0.99$ (Fig. 3.5). Note that we typically report reaction rates relative to $k^{obs}_{cat}\,(WT)$, the observed total reaction rate of the wild-type sequence: $\tilde{k}_{cat,i}\,(S_j) = k_{cat,i}\,(S_j)\,/k^{obs}_{cat}\,(WT)$ and $\tilde{k}^{obs}_{cat,i}\,(S_j) = k^{obs}_{cat,i}\,(S_j)\,/k^{obs}_{cat}\,(WT)$ are the predicted and observed relative reaction rates for product i.

Since the range and the average of product-specific reaction rates vary widely depending on the product type (Fig. 3.2), we expect that, for a particular product, the complexity of the model and, correspondingly, the number of non-zero one-body terms and two-body couplings will be correlated with the number of enzyme variants capable of making that product. Indeed, we find that the model complexity is the highest for the original BFS product, (E)-β-farnesene, which is produced at non-zero rate by most enzymes. This is not surprising since free energy landscapes for a given product that are based on just a few enzyme variants with detectable output should be easier to model and require fewer fitting parameters.

We observe that at 24 out of 25 positions under consideration (position 559 being the sole exception), mutating a residue results in a change in one or, more typically, several single-aa contributions to product-specific free energy landscapes (Fig. 3.6). Interestingly, changes in G_3 are predominantly negative, meaning that the mutations tend to have adverse effects on the overall values of reaction rates (Eq. (3.2)). The only exception to this rule is position 402. A Y402L mutation at this position does not just increase the total reaction rate due to the product-independent lowering of the free energy barrier, it also rebalances the enzyme specificity towards the cyclic products, by lowering the relative reaction rates for linear products (E)-β-farnesene (the original BFS product) and nerolidol and increasing the relative reaction rates for cyclic products zingiberene, α-bisabolene, and α-bisabolol (Fig. 3.6). Thus 402 plays a role of a "gateway" cyclization-unlocking mutation between enzymes producing linear and cyclic products. Other key positions which promote production of cyclic products are 324 and 429. Finally, note that mutations at 10 out of 25 positions result in single-aa terms that suppress (E)-β-farnesene production.

In addition to single-aa terms, the structure of the free energy landscapes is shaped by 54 non-zero coupling terms between pairs of aa positions, 48 of which affect $G_{4,i}$ values and the other 6 correspond to G_3 (Fig. 3.7A). Interestingly, most of the $G_{4,i}$ non-zero couplings contribute to a single free energy landscape corresponding to (E)-β-farnesene – the original linear product of the wild-type BFS enzyme. Out of the 48 two-aa $G_{4,i}$ terms which determine enzyme specificity, 10 increase reaction rates for cyclic products and only 2 decrease those rates; for linear products, 21 terms contribute to a rate increase and 15 to a rate decrease. Overall, the values tend to be more negative for cyclic products (Fig. 3.7B), meaning that, as a rule, two-body terms tend to promote cyclization. As shown in Fig. 3.7C, only 8 aa pairs affect more than one free energy

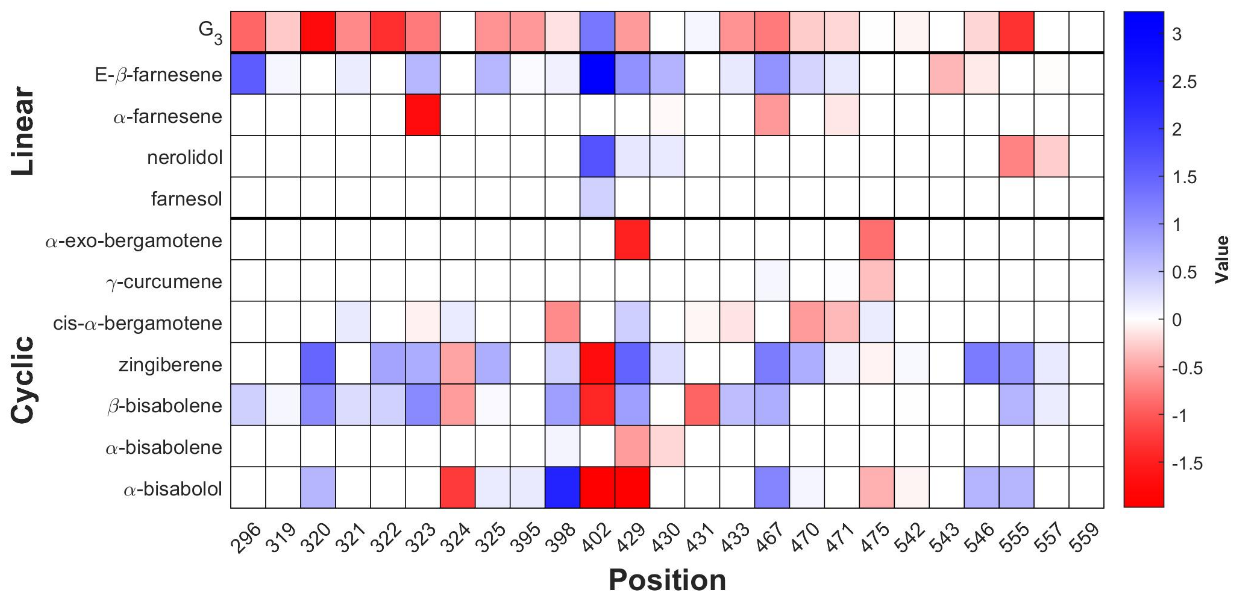


Figure 3.6: **One-aa contributions to Michaelis-Menten free energies.** Fitted values of one-aa model parameters $H_k(M)$ (G_3) and $h_k^{(i)}(M)$ $(G_{4,i})$

landscape: for example, the 398-429 coupling simultaneously increases the reaction rates of cyclic products α-exo-bergamotene and zingiberene. Correspondingly, 37 aa pairs have a single non-zero coupling term and therefore mutations at these positions affect only one free energy landscape. The remaining 93 pairs of positions do not contribute to the free energies at all.

Finally, we have demonstrated the predictive power of the pairwise model by testing its ability to predict reaction rates of novel enzyme sequences, after training the model on only a part of the available data. Specifically, we have randomly chosen 82 enzyme sequences and trained the model with the LASSO constraint and cross-validation as described above, using the $\tilde{k}^{obs}_{cat,i}$ values corresponding to those sequences as input. The model was subsequently used to predict the $\tilde{k}_{cat,i}$ values for the remaining 40 enzyme sequences which were not used in training the model, with $R^2 = 0.92$ (Fig. 3.8A). Since the prediction is dominated by several datapoints with larger values of kinetic rates, we have also examined the distribution of the differences between predicted and observed kinetic rate values (Fig. 3.8B) and the R^2 values of partial datasets obtained by sorting the set of observed kinetic rates by magnitude (Fig. 3.8C). We find that the prediction

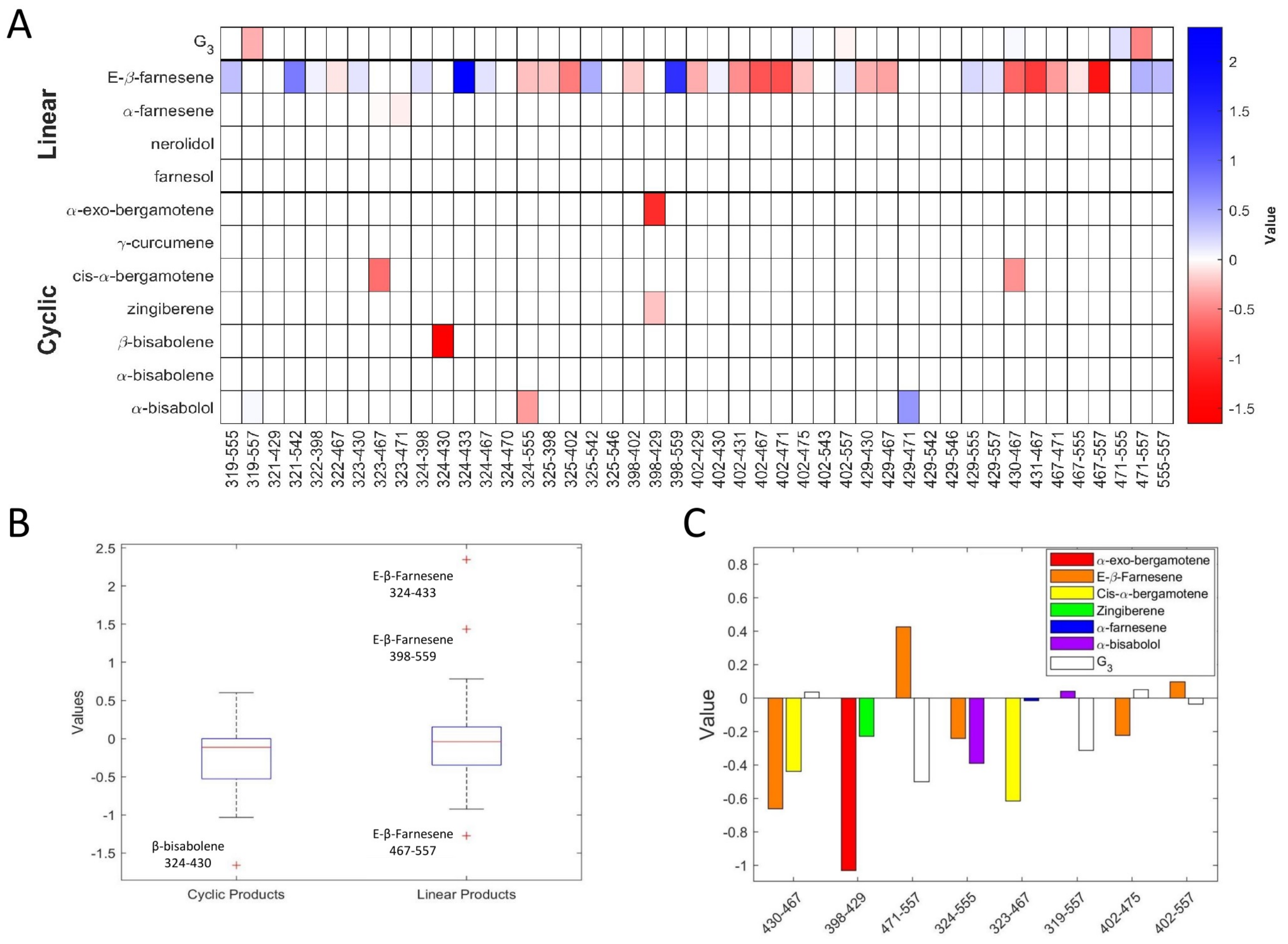


Figure 3.7: **Two-aa contributions to Michaelis-Menten free energies.** (A) Fitted values of two-aa model parameters $E_{kl}(M,M)$ (G_3) and $\varepsilon_{kl}^{(i)}(M,M)$ ($G_{4,i}$). (B) Box-and-whiskers plots (created using the boxplot function with default parameters in MATLAB) for the distributions of two-aa model parameters for linear and cyclic enzyme products. (C) Two-aa model parameters $E_{kl}(M,M)$ (G_3) and $\varepsilon_{kl}^{(i)}(M,M)$ ($G_{4,i}$) at 8 pairs of positions where two or more two-aa contributions adopt non-zero values.

errors are in fact higher for the outliers. However, as expected, for very low kinetic rates the errors become comparable to the predicted values themselves. As a result, the R^2 values are low until some of the large-value outliers are included into the dataset.

3.4 Significance of Biophysical Model

3.4.1 Structure of Michaelis-Menten free energy landscapes

The sparseness of the free energy models described above translates into Michaelis-Menten free energy landscapes with simple and interpretable structure. To illustrate this point, we first focus on the G_4 landscape for the cyclic product -bisabolol (Fig. 3.9,3.10,3.11). In the combined library, this landscape is controlled by 14 one-aa and 3 two-aa model parameters; among one-aa parameters, 5 are above $1k_BT$, and 3 of those, at positions 324, 402, and 429, enhance relative reaction rates for -bisabolol by lowering the G_4 barrier (Fig. 3.9). In comparison to these leading one-aa contributions, two-aa terms play a secondary role. Consequently, in the M5 library, where the amino acid mutations are restricted to positions 319, 402, 429, 555, and 557, the structure of the free energy landscape is largely determined by the states of amino acids at positions 402 and 429 (a third position, 555, plays a secondary role) (red bars in Fig. 3.9). Thus, the G_4 landscape is divided into 4 distinct sectors, with the wild-type BFS sequence (TYTHE at the 5 variable positions) being $\approx 3.2k_BT$ less favorable for α-bisabolol production than the 5-point mutant, ALGRG (Fig. 3.10). However, other sequences in the same cluster, such as ALGHG and TLGHG, are characterized by even lower G_4 barriers. In fact, as noted above, it is sufficient to carry out just two mutations, Y402L and T429G, in order to obtain an α-bisabolol producing enzyme.

Although the specificity of a given enzyme is controlled by the relative heights of the G_4 barriers for each product, its overall output also depends on the height of the G_3 barrier which we have assumed to be independent of the product type. Similar to the G_4 landscape for α-bisabolol , the G_3 free energy landscape in the combined library is a function of just 20 one-aa and 5 two-aa model parameters, with only 4 one-aa parameters, at positions 320, 322, 555, and 402, above $1k_BT$ (Fig. 3.12). Two

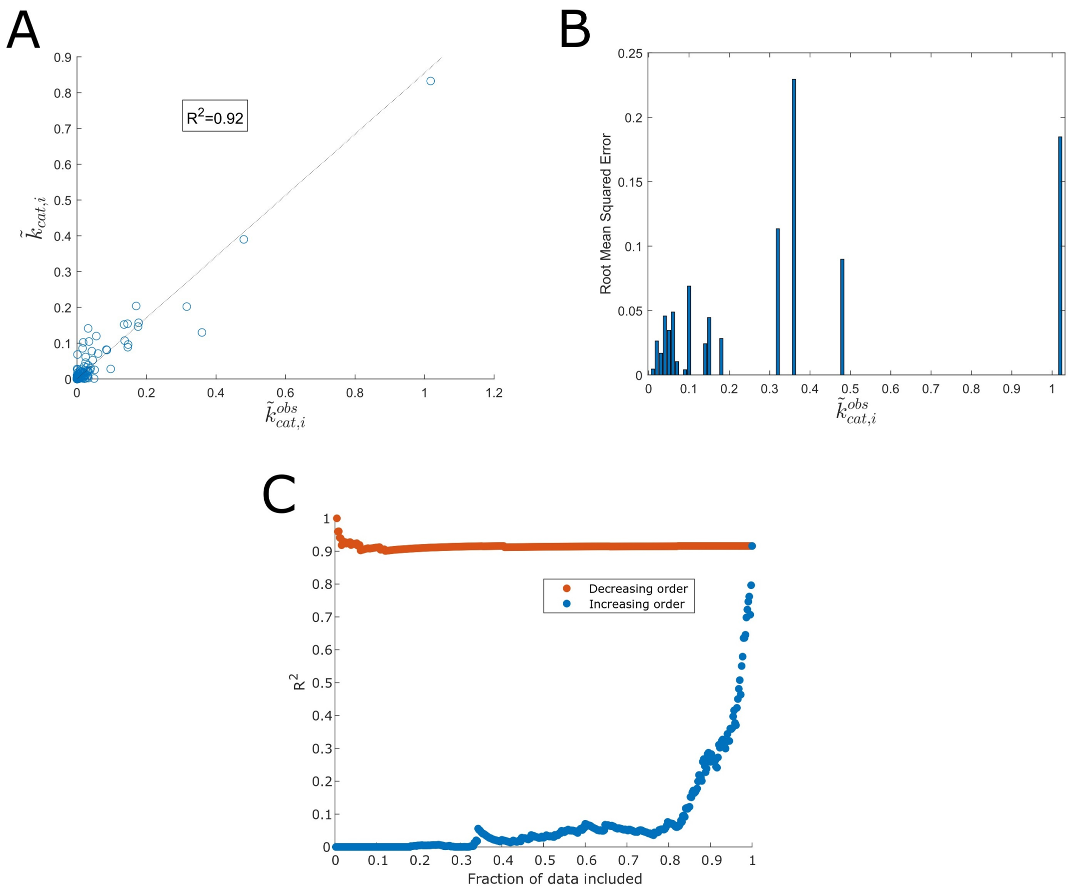


Figure 3.8: **Prediction of novel reaction rates.** (A) Comparison of predicted and observed reaction rates $k_{cat,i}$ for 40 enzyme sequences which were not used in training the model. The parameters of the model were obtained by LASSO with cross-validation, using reaction rates for the other 82 enzyme sequences as input data. (B) Root-mean-square error between predicted and observed product-specific kinetic rates. All $\tilde{k}^{obs}_{cat,i}(S_j)$ values in the 40-sequence test set were divided into 102 bins of width 0.01. For $\tilde{k}^{obs}_{cat,i}(S_j)$ values in each non-empty bin, shown is the root-mean-square error with respect to predicted rates $\tilde{k}^{obs}_{cat,i}(S_j)$. (C) Shown are partial R^2 values for the 40-sequence test set in which all data points were sorted by the magnitude of the $\tilde{k}^{obs}_{cat,i}(S_j)$ values. Blue: partial R^2 values vs. fraction of all data points in the test set, starting from the lowest values. Red: partial R^2 values vs. fraction of all data points in the test set, starting from the highest values.

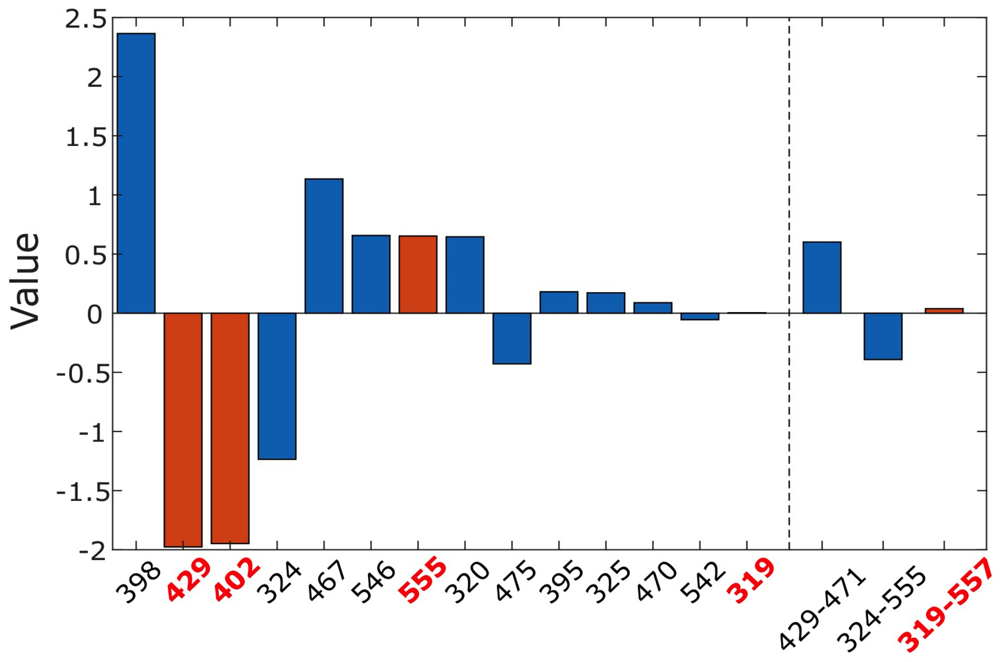


Figure 3.9: The values of all one-aa and two-aa non-zero parameters in the G_4 pairwise expansion for α-bisabolol fitted to the combined library data. Positions and position pairs that occur in the M5 library are highlighted in red

of these positions, 402 and 555, are variable in the M5 library and hence the amino acid states at these positions largely determine the structure of the G_3 free energy landscape (red bars in Fig. 3.12 and Fig. 3.13; positions 429 and 319 play a secondary role). Interestingly, although positions 319 and 557 are characterized by small (319) and zero (557) one-aa contributions, they shape the free energy landscape through a two-aa term. We observe that the 5-point mutant, ALGRG, is lower in the overall output than the wild-type sequence, TYTHE: the corresponding G_3 value is lower by $\approx 1.2k_BT$, which makes the kinetic barrier higher overall. The Y402L mutation is the only major contributor to lowering the free energy barrier (Fig. 3.12); consequently, the TLGHE double mutant discussed above (Y402L/T429G) favors both α-bisabolol production and high overall output.

To investigate the effect of these mutations on other products, we have considered G_4 free energy landscapes for the wild-type linear product, (E)-β-farnesene (Fig. 3.15, 3.16, 3.17), and another cyclic product of practical importance, zingiberene (Fig. 3.18,

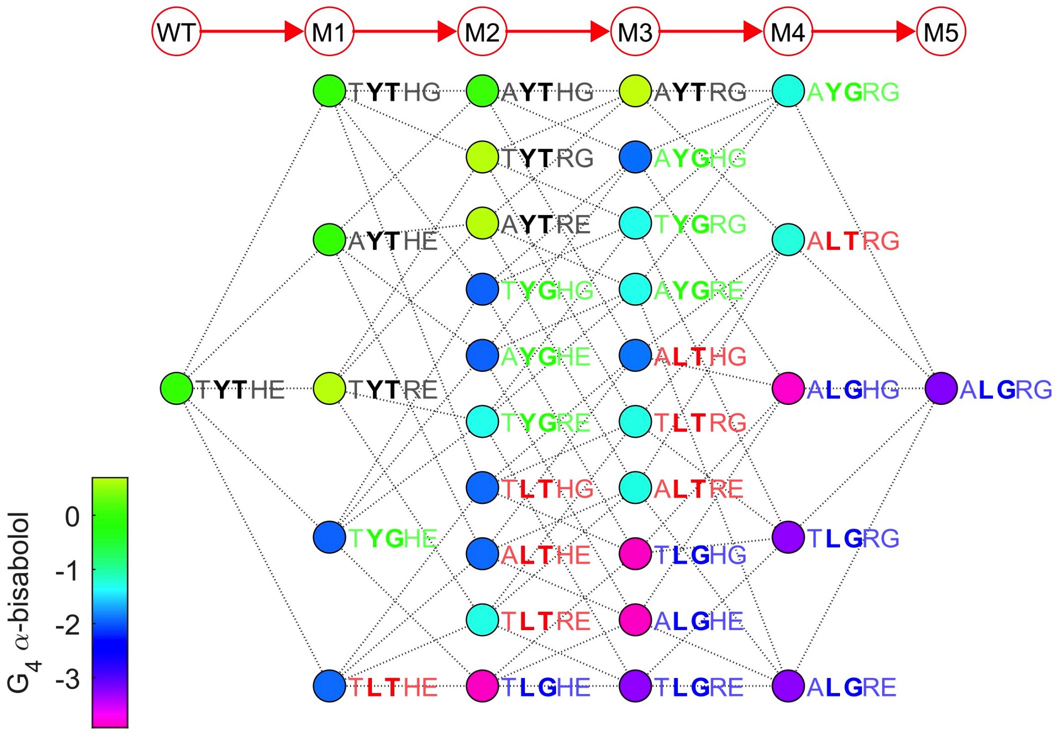


Figure 3.10: The free energy landscape for α-bisabolol G_4 values on the M5 library, which contains all combinations of mutant and wild-type amino acids at positions 319, 402, 429, 555 and 557. Each node on the landscape is labeled by a string of amino acids at the 5 positions. Nodes that differ by a single amino acid substitution are connected by an edge. The arrows and circles above the landscape indicate the number of mutations away from the wild-type *A. annua* BFS sequence TYTHE. Each node is colored according to the G_4 value for α-bisabolol. In each column, sequences are sorted according to the values of one-aa contributions at positions 402, 429 and 555, which are the 3 largest among the 5 positions considered (Fig. 3.6). From top to bottom, the sequences with a given number of mutations with respect to the wild-type sequence are sorted in the following order: WWW, WWM, WMW, WMM, MWW, MWM, MMW, MMM. Sequences which differ only at positions 319 and 557 (if any) appear in the order W...W, W...M, M...W, M...M. All nodes are sorted into 4 clusters on the basis of amino acids at positions 402 and 429 which have the largest one-aa contributions (red bars in Fig 3.9). These positions are highlighted in bold in each sequence; all sequences are color-coded according to their cluster assignments.

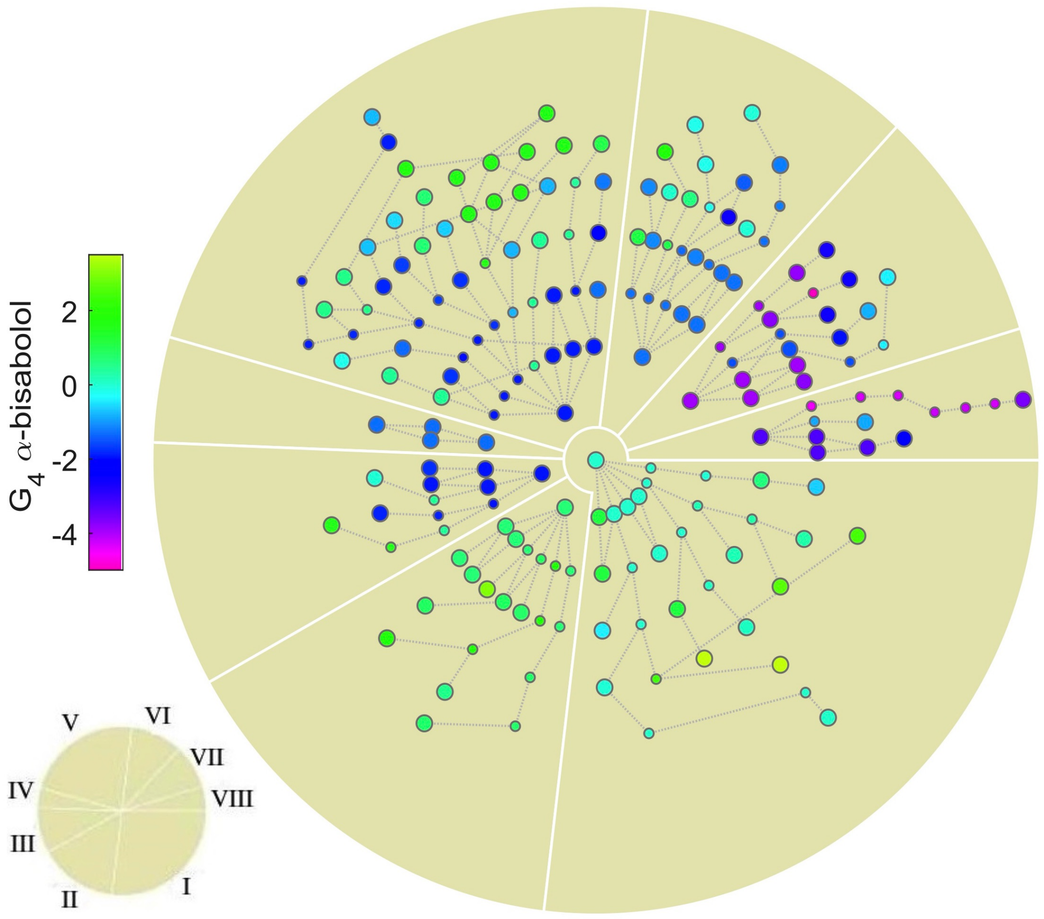


Figure 3.11: Predictions of G_4 for α-bisabolol on the combined library. All nodes are arranged in circles according to the number of mutations away from the wild-type *A. annua* BFS sequence. Nodes are clustered on the basis of positions 402, 429 and 555 in the order WWW, WWM, WMW, WMM, MWW, MWM, MMW, MMM for clusters I-VIII, respectively. These three positions were chosen since they have the highest position score: $P_i = \left|h_i^{\alpha-bisabolol}\right| + \sum_{j\neq i} |\varepsilon_{ij}^{\alpha-bisabolol}| + |H_i| + \sum_{j\neq i} |E_{ij}|$, which represents the sum of the absolute magnitudes of all one-aa and two-aa model parameters associated with position i. Nodes in the same cluster are connected by an edge if their sequences differ by a single amino acid substitution. Within each cluster and each circular shell, nodes are sorted so as to minimize the number of edges crossing each other. Large circles denote sequences in the combined dataset, whereas small circles show sequences for which G_4 values were predicted. Each node is colored according to the G_4 value for α-bisabolol.

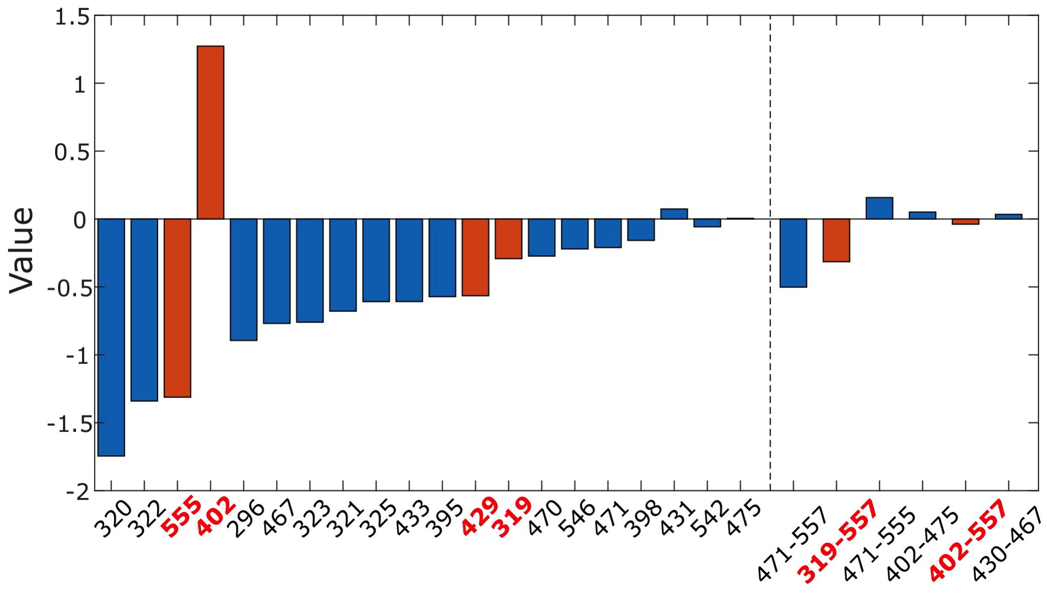


Figure 3.12: The values of all one-aa and two-aa non-zero parameters in the G_3 pairwise expansion fitted to the combined library data. Positions and position pairs that occur in the M5 library are highlighted in red

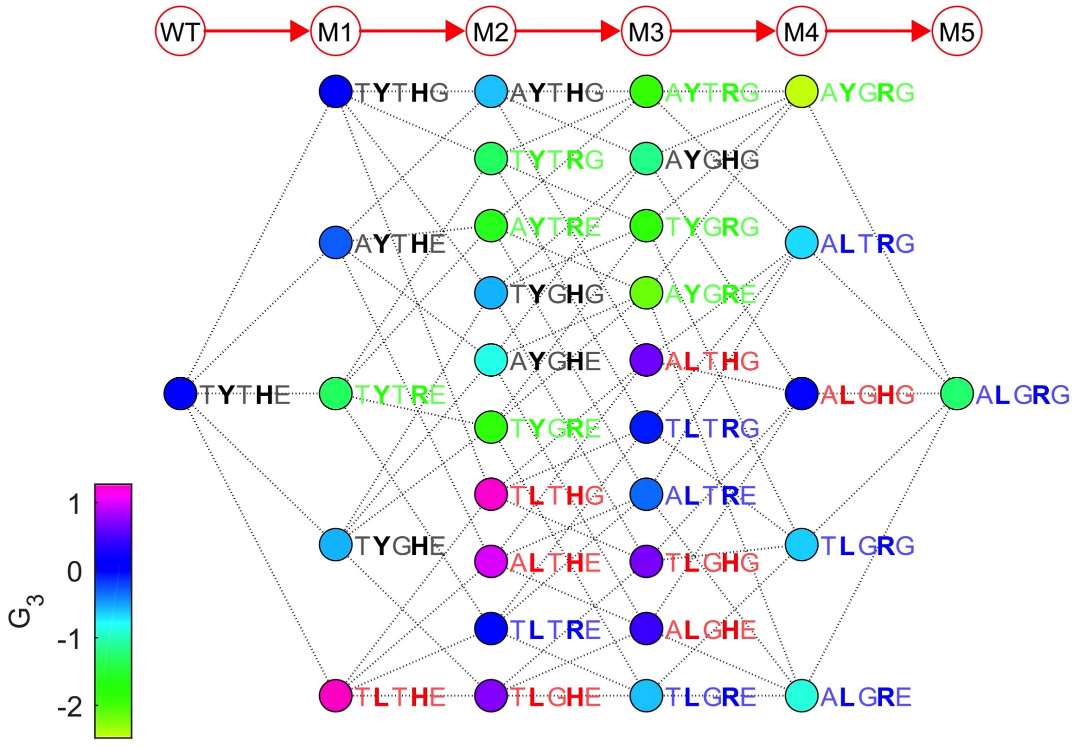


Figure 3.13: The G_3 free energy landscape values on the M5 library same as Fig. 3.10

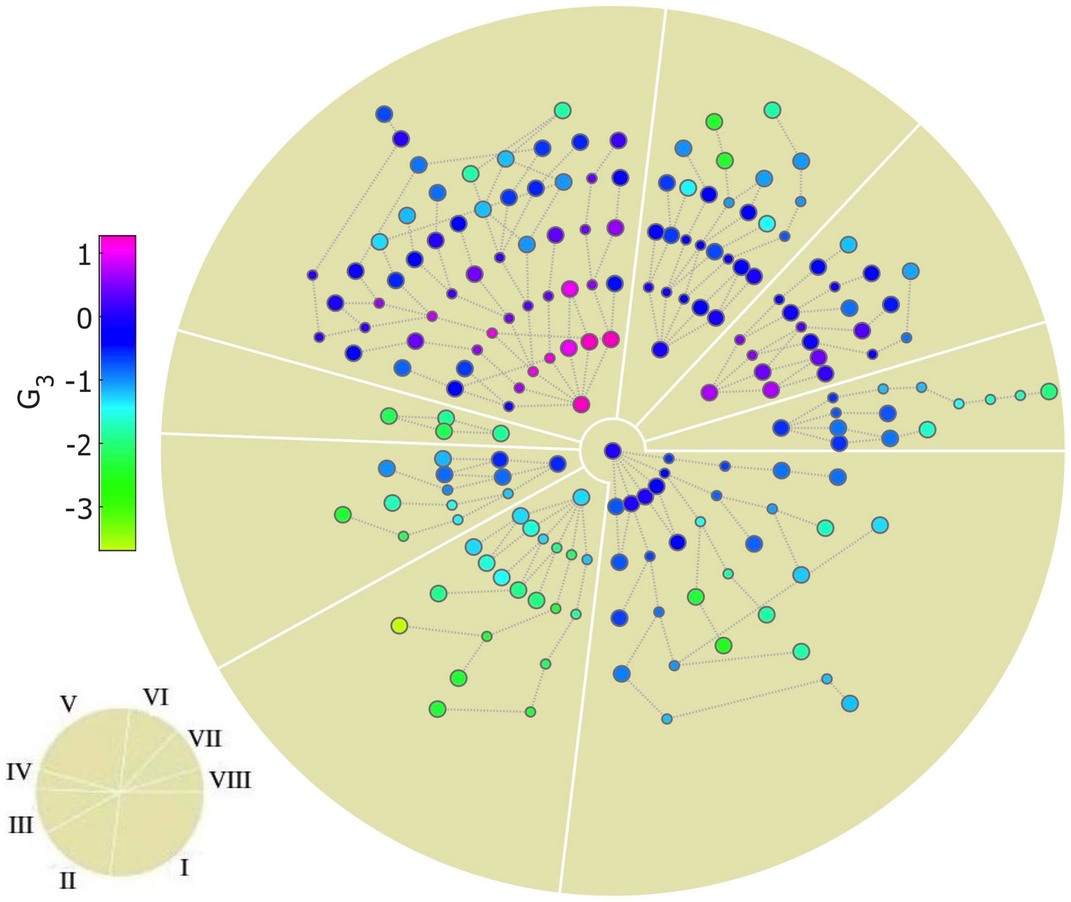


Figure 3.14: Predictions of G_3 free energy landscape on the combined library same as Fig. 3.11

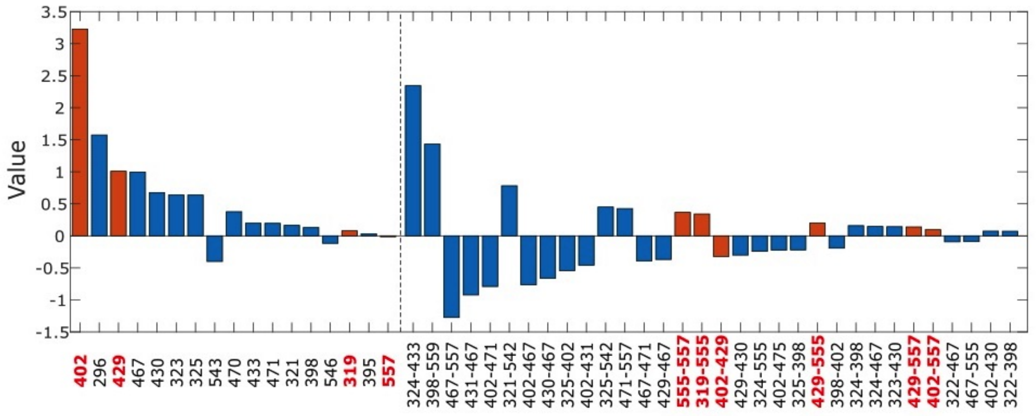


Figure 3.15: The values of all one-aa and two-aa non-zero parameters in the G_4 pairwise expansion for E-β-farnesene fitted to the combined library data. Positions and position pairs that occur in the M5 library are highlighted in red

3.19, 3.20). Interestingly, the (E)-β-farnesene landscape is characterized by 17 one-aa and 28 two-aa terms and thus can be expected to be more epistatic (Fig. 3.15), although its projection onto M5 sequences is fairly sparse, being mainly determined by aa states at positions 402 and 429 (Fig. 3.16). As expected, the TLGHE double mutant (Y402L/T429G) and especially the 5-point mutant, ALGRG, are characterized by sharply decreased levels of (E)-β-farnesene production. Correspondingly, the best (E)-β-farnesene producing enzymes are those with aa at positions 402 and 429 left in the wild-type state. The G_4 free energy landscape for zingiberene is determined almost exclusively by one-aa contributions (Fig. 3.18), of which aa states at positions 402, 429, and 555 structure the landscape's projection onto M5 sequences. Since the Y402L mutation is the only one favorable for zingiberene production, sequences in the "LT" cluster (shown in red in Fig. 3.19), which includes a single mutant TLTHE, are best zingiberene producers.

It is also informative to consider the free energy landscapes on all sequences from the combined library. In Fig. 3.11, nodes are arranged radially around the wild-type sequence according to the number of mutations. Sequences are clustered into 8 sectors on the basis of aa states at positions 402, 429 and 555, which contribute the most when both one-aa and two-aa terms are taken into account. In addition to 122

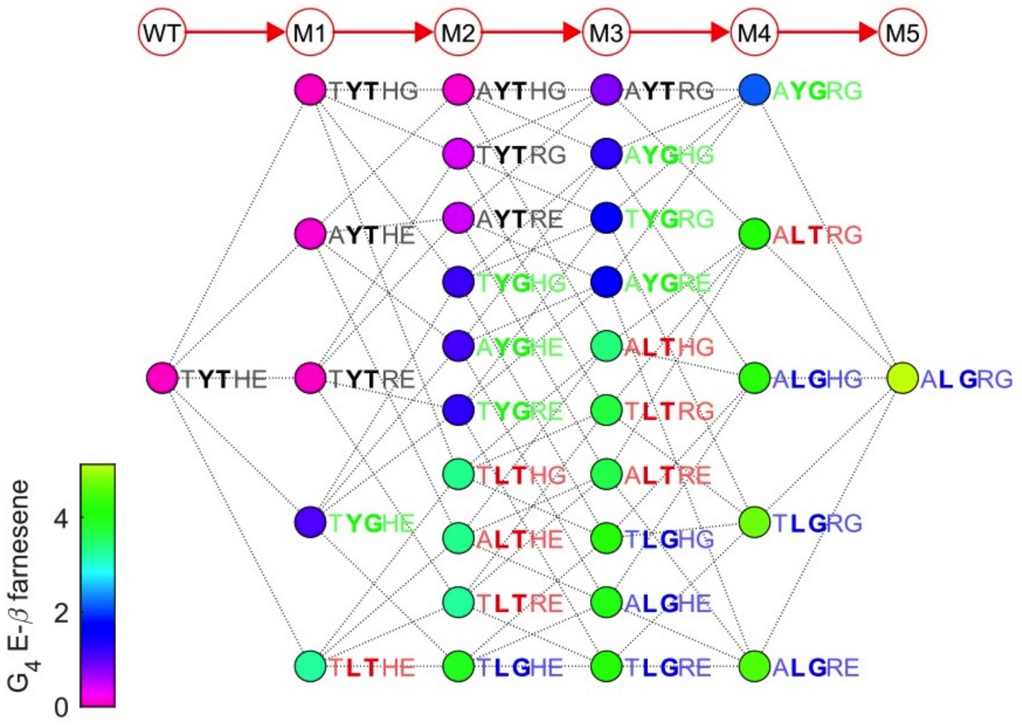


Figure 3.16: The free energy landscape for E-β-farnesene G_4 same as Fig. 3.10

sequences in the combined dataset, we have made predictions for 69 additional sequences which were chosen to fill in the gaps in mutational pathways connecting experimentally characterized sequences. Clusters VII and VIII, which have both Y402L and T429G mutations (cluster VII has H whereas cluster VIII has R at position 555), are enriched the most in α-bisabolol producing enzymes (Fig. 3.11). These clusters, along with clusters V and VI, tend to contain sequences that are least favorable for (E)-β-farnesene production due to the Y402L mutation (Fig. 3.17). For zingiberene, the most favorable sequences are concentrated in cluster V, although the G_4 free energy barrier is rarely lowered by more than $1k_BT$ (Fig. 3.20). Finally, sequences with higher values of G_3 (which is beneficial for the overall output) tend to be found in clusters V and VII (Fig. 3.14). In summary, sequences in cluster VII (Fig. 3.11,3.14) are the best candidates for α-bisabolol production in the combined library; specific candidates can be chosen based on how critical it is to produce α-bisabolol specifically (as opposed e.g. to a mixture of α-bisabolol , zingiberene, and other cyclic products).

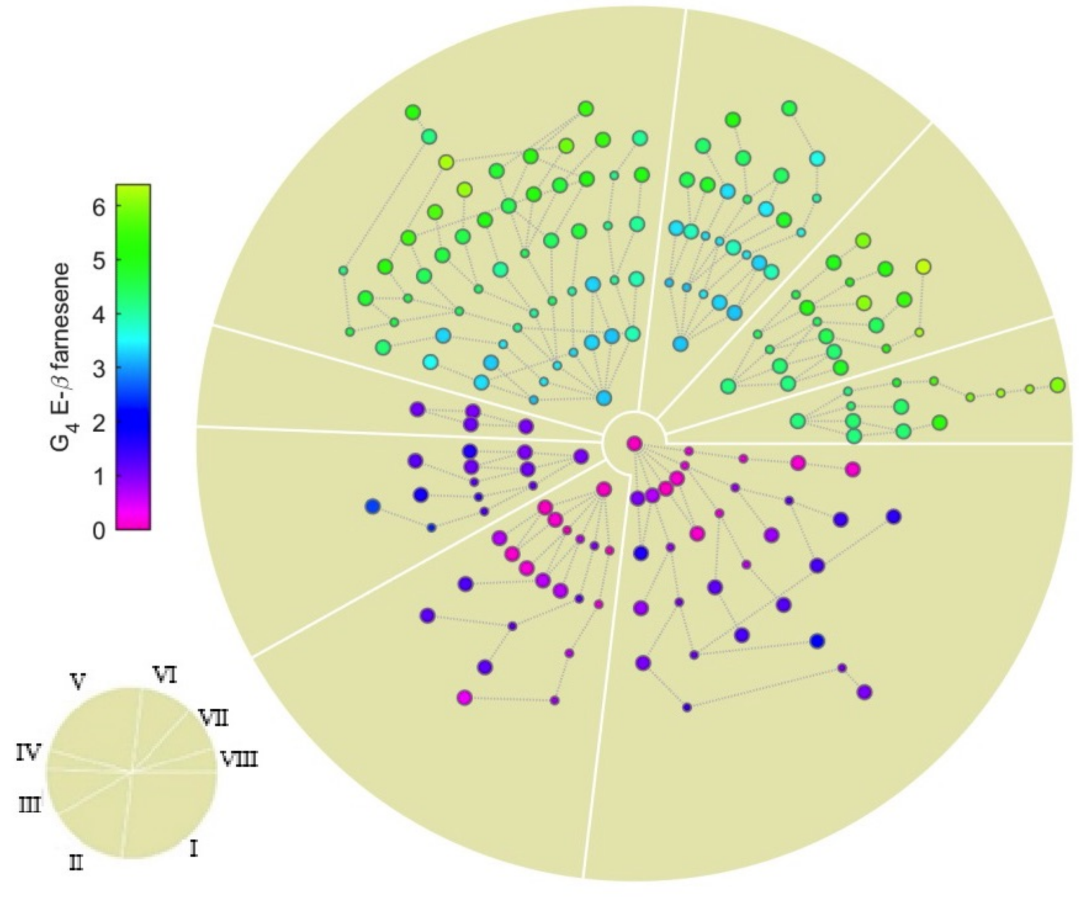


Figure 3.17: Predictions of G_4 for E-β-farnesene same as Fig. 3.11

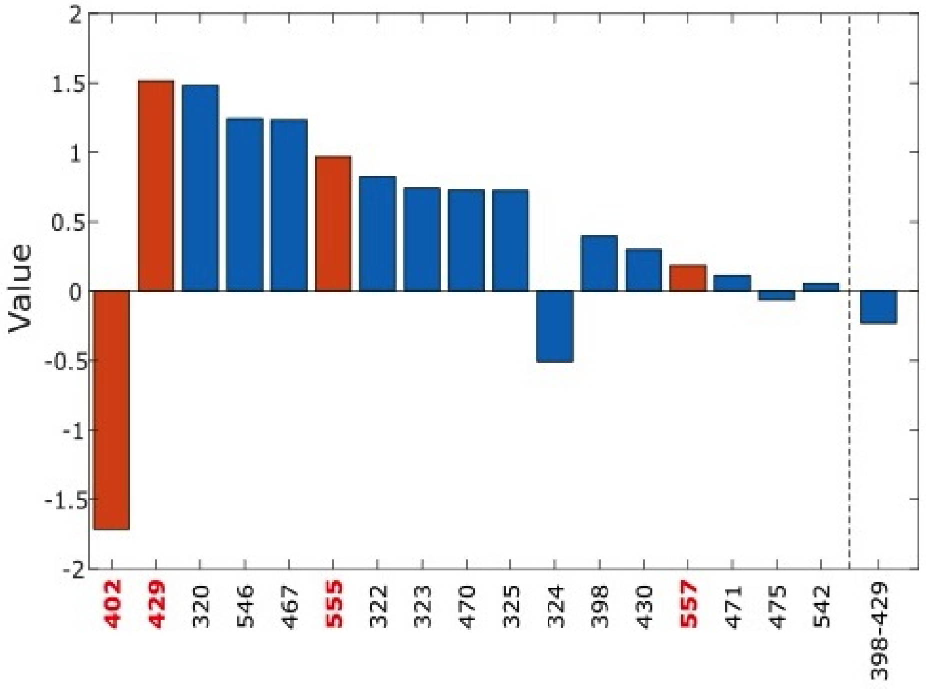


Figure 3.18: The values of all one-aa and two-aa non-zero parameters in the G_4 pairwise expansion for zingiberene fitted to the combined library data. Positions and position pairs that occur in the M5 library are highlighted in red

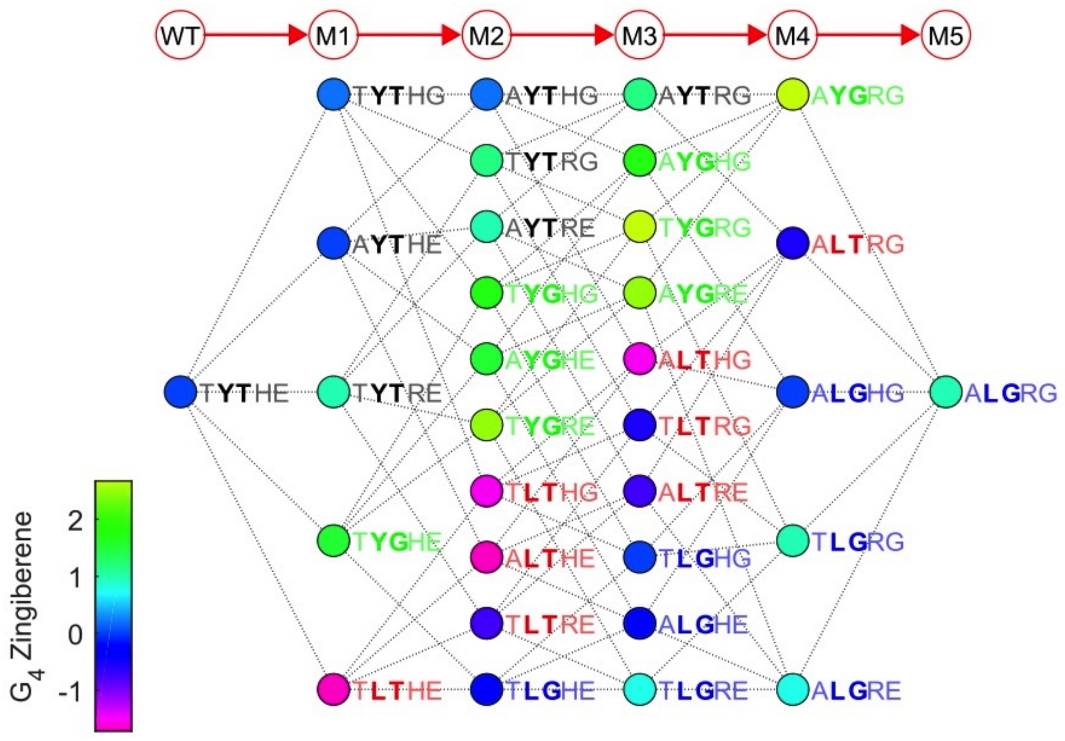


Figure 3.19: The free energy landscape for zingiberene G_4 same as Fig. 3.10

3.4.2 Epistasis on free energy landscapes

The notion of epistasis is closely related to the extent of non-linearity and ruggedness observed in fitness or energy landscapes ([22], [23], [24]). In its most basic form, epistasis involves aa states at two distinct positions in the sequence. In the case of two aa states (such as the W and M states), epistatic interactions allow for a simple geometric interpretation (Fig. 3.21). Note that the no-epistasis scenario implies the absence of an energetic or a fitness coupling between the two sites, while the other three scenarios (magnitude, sign or reciprocal sign epistasis) are controlled by the magnitude and the sign of the relevant coupling terms. In our case, only the MM coupling can be non-zero by construction; however, the two aa sites in question are embedded into longer sequences and therefore the amount and the type of epistasis may also be affected by the two-aa terms in which one of the partners is outside of the current pair.

Since the magnitude and the sign of epistasis between two aa sites depend on the rest

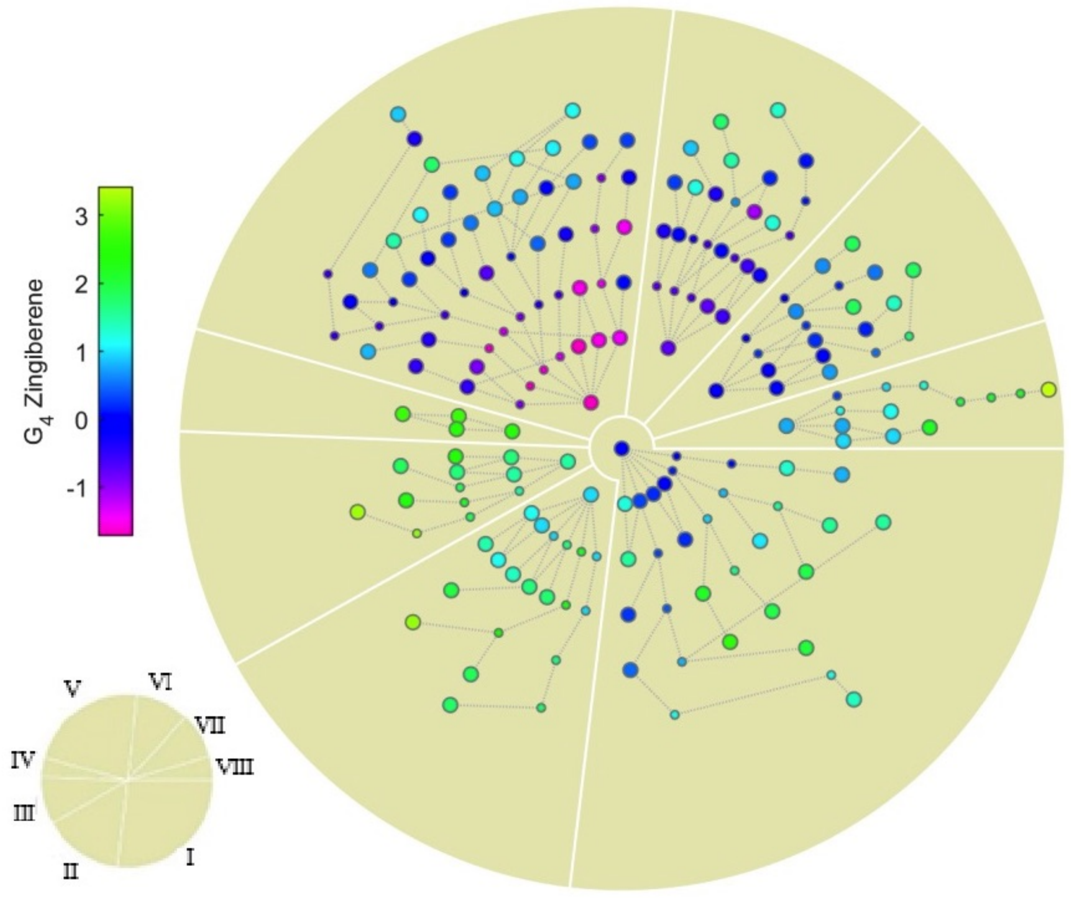


Figure 3.20: Predictions of G_4 for zingiberene same as Fig. 3.11

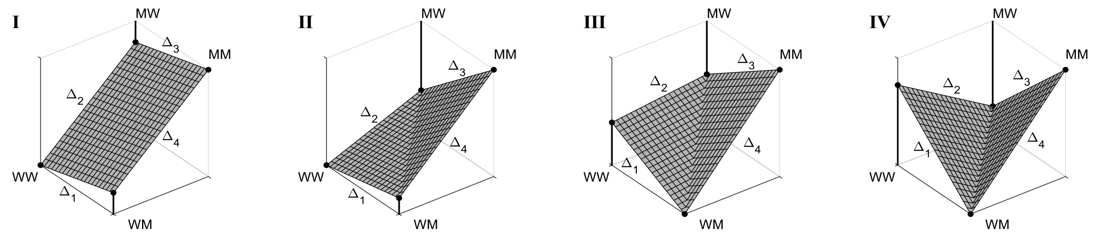


Figure 3.21: **The four types of epistasis in a two-site system.** The aa at each site can be in either W or M state. Panel I: no epistasis, with each mutation contributing the same amount to the total free energy (or fitness) regardless of the aa state at the other site. Panel II: magnitude epistasis, where the magnitude (but not the sign) of the aa free energy (or fitness) contribution depends on the aa state at the other site. Panel III: sign epistasis, where for one of the sites, the sign (and, in general, the magnitude) of the aa free energy (or fitness) contribution depends on the aa state at the other site. Panel IV: reciprocal sign epistasis, where the aa free energy (or fitness) contribution at both sites depends on the aa state at the other site.

of sequence, we have focused our attention on the subsets of sequences which are identical outside of the two positions i and j for which epistatic interactions are computed. At these positions, data has to be available for all 4 aa states: (W, W),(W, M),(M, W) and (M, M). These requirements result in 4 sequence subsets of the same size: $T_{ij}\left(W, W\right) = [S_1^{(1)}, S_2^{(1)}, \ldots, S_n^{(1)}]$, $T_{ij}\left(W, M\right) = [S_1^{(2)}, S_2^{(2)}, \ldots, S_n^{(2)}]$, $T_{ij}\left(M, W\right) = [S_1^{(3)}, S_2^{(3)}, \ldots, S_n^{(3)}]$, and $T_{ij}\left(M, M\right) = [S_1^{(4)}, S_2^{(4)}, \ldots, S_n^{(4)}]$ (Fig. 3.22). Next, we compute the free energies $\overline{G}_{ij}\left(W, W\right)$, $\overline{G}_{ij}\left(M, W\right)$, $\overline{G}_{ij}\left(W, M\right)$, and $\overline{G}_{ij}\left(M, M\right)$ which are simply the G_3 or G_4 free energy values averaged over all sequences in the corresponding T_{ij} subset. Finally, we define the differences of the averaged free energies along each edge in the geometric shapes of Fig. 3.21: $\Delta_1 = \overline{G}_{ij}\left(W, M\right) - \overline{G}_{ij}\left(W, W\right)$, $\Delta_2 = \overline{G}_{ij}\left(M, W\right) - \overline{G}_{ij}\left(W, W\right)$, $\Delta_3 = \overline{G}_{ij}\left(M, M\right) - \overline{G}_{ij}\left(M, W\right)$, and $\Delta_4 = \overline{G}_{ij}\left(M, M\right) - \overline{G}_{ij}\left(W, M\right)$. Then the four epistasis types can be succinctly summarized as follows: $\Delta_1 = \Delta_3$ and $\Delta_2 = \Delta_4$ correspond to the absence of epistasis, otherwise $sgn(\Delta_1) = sgn(\Delta_3)$ and $sgn(\Delta_2) = sgn(\Delta_4)$ represent magnitude epistasis, $sgn(\Delta_1) \neq sgn(\Delta_3)$, $sgn(\Delta_2) = sgn(\Delta_4)$ or $sgn(\Delta_1) = sgn(\Delta_3)$, $sgn(\Delta_2) \neq sgn(\Delta_4)$ represent sign epistasis (the two cases correspond to two opposite pairs of edges in Fig. 3.21, panel III), and $sgn(\Delta_1) \neq sgn(\Delta_3)$, $sgn(\Delta_2) \neq sgn(\Delta_4)$ correspond to the reciprocal sign epistasis. Note that on fitness landscapes, sign epistasis can significantly affect genotype accessibility by making some evolutionary trajectories unavailable or unlikely ([22]), whereas reciprocal sign epistasis is a necessary condition for the existence of multiple local maxima ([25]).

Next, we define an epistatic score ES_{ij} as the absolute magnitude of the difference between the Δ values on the two pairs of opposite edges in the geometric shapes shown in Fig. 3.21:

$$ES_{ij} = \left|\Delta_4 - \Delta_2\right| = \left|\Delta_3 - \Delta_1\right|, \tag{3.8}$$

such that $ES_{ij} = 0.0$ in the case of no epistasis, and positive otherwise.

Consistent with the above discussion of Michaelis-Menten landscape structure and appearance, all free energy landscapes are characterized by a limited amount of epistasis. Indeed, in our combined dataset we find only 16 pairs of positions i and j for which the above analysis of epistatic interactions can be carried out: $\binom{5}{2} = 10$ pairs

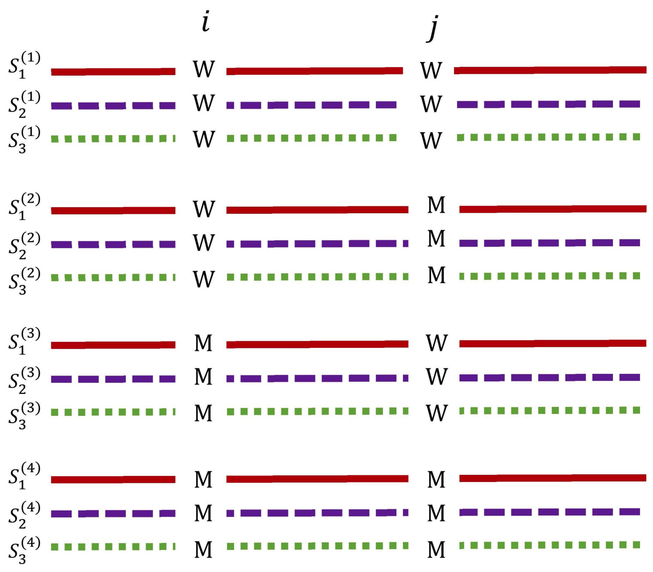


Figure 3.22: **Schematic representation of a set of sequences $S_k^{(a)}$ divided into 4 subsets $T_{ij}(A, B)$ of equal size, where $A, B = [W, M]$.** The sequence subsets are used to calculate the epistasis score ES_{ij} for aa positions i and j, as described in the text. Note that outside of the positions i and j, sequences in each subset are exactly the same. (C) Plot of the free energy epistasis scores ES_{ij} vs. the $C_\alpha - C_\alpha$ spatial distances (in) between aa positions i and j. All epistasis scores with magnitudes < 0.01 were excluded from the plot. A vertical dashed grey line at 15.8 shows the average $C_\alpha - C_\alpha$ distance for all 138 aa pairs.

come from the M5 library, with $2^3 = 8$ sequences in each T_{ij} subset, 1 pair (402-430) has 3 sequences in each T_{ij} subset, and 5 more pairs occur on the background of a single sequence. Since there are 12 distinct free energy landscapes, we have 192 potentially epistatic instances in our dataset. Out of those, only 15 pairs are characterized by $ES_{ij} > 0.01k_BT$, with 13 exhibiting magnitude epistasis, 1 showing sign epistasis (positions 430-467 on the G_3 landscape, with $ES_{\text{430-467}} = 0.03k_BT$), and 1 more demonstrating reciprocal sign epistasis (positions 430-467 on the G_4 landscape for cis–bergamotene, with $ES_{\text{430-467}} = 0.44k_BT$). Interestingly, 10 out of 13 pairs with magnitude epistasis occur on the (E)-β-farnesene landscape, and 2 other instances occur on the G_3 landscape. Thus, (E)-β-farnesene production mediated by a wild-type enzyme is characterized by significantly more pronounced epistatic interactions than production of cyclic terpenes by the enzymes in our mutant library.

To investigate whether higher levels of free energy epistasis occur in pairs of residues that are close to each other in 3.12 space, we have plotted the 15 pairs of residues with non-zero ES_{ij} scores vs. the corresponding $C_\alpha - C_\alpha$ distances in Fig. 3.23. Although the two residues in pairs with top 3 ES_{ij} values do tend to be closer to each other than 15.8 , the average distance between all 138 pairs under consideration (cf. the vertical dashed line in Fig. 3.23), the overall trend is rather weak, especially if all pairs that are close to each other in the linear sequence, such as 555-557, are excluded from the consideration. For example, the 319-555 pair on the (E)-β-farnesene landscape exhibits significant magnitude epistasis, despite the fact that these residues are separated by more than 20 Å. Remarkably, the 430-467 pair appears as epistatic on 3 free energy landscapes, with the corresponding ES_{ij} scores ranked 1, 3, and 15 by absolute magnitude (out of 15 pairs with $ES_{ij} > 0.01k_BT$.

Strong epistasis between positions 430 and 467 can be readily rationalized by their spatial proximity in the BFS structural model (Fig. 3.24A). The residues in the 430-467 pair are within van der Waals distance ($< 3\mathring{A}$) from each other and are located at the bottom of the active site, making direct contacts with the isopropenyl tail of the substrate FPP. As such, substitutions at these positions are expected to be interdependent, with hydrophobic contacts likely accounting for the physical interactions between the

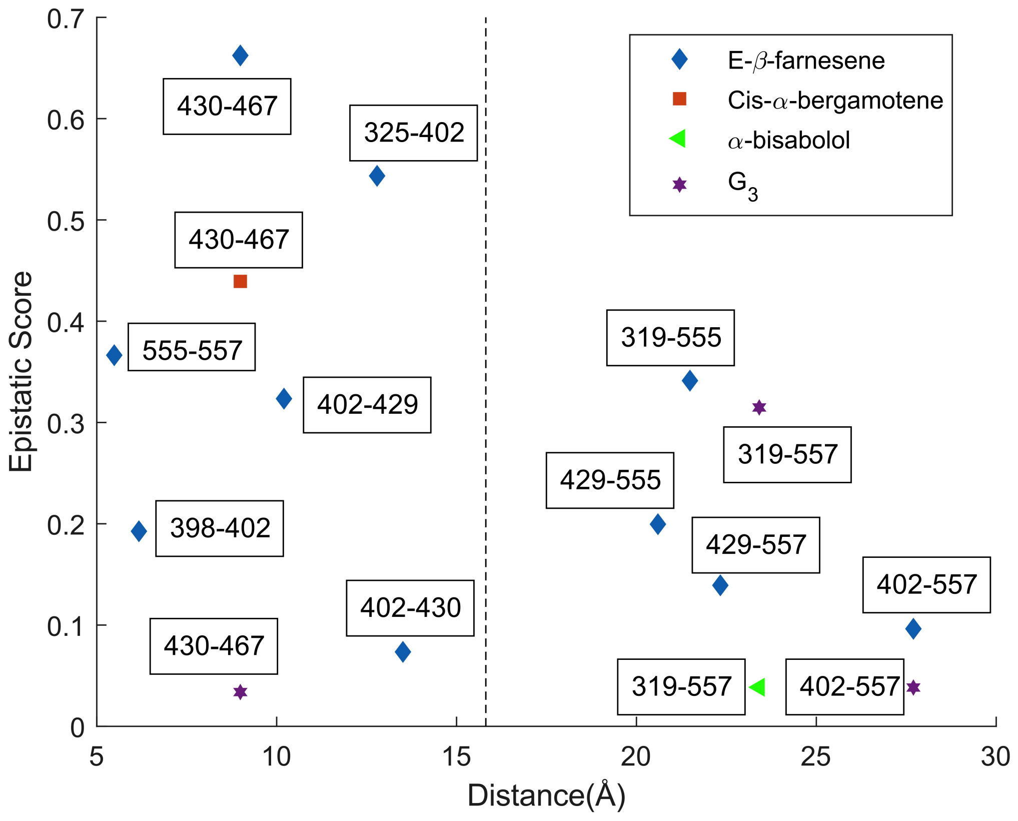


Figure 3.23: Plot of the free energy epistasis scores ES_{ij} vs. the $C_\alpha - C_\alpha$ spatial distances (in) between aa positions i and j. All epistasis scores with magnitudes < 0.01 were excluded from the plot. A vertical dashed grey line at 15.8 shows the average $C_\alpha - C_\alpha$ distance for all 138 aa pairs.

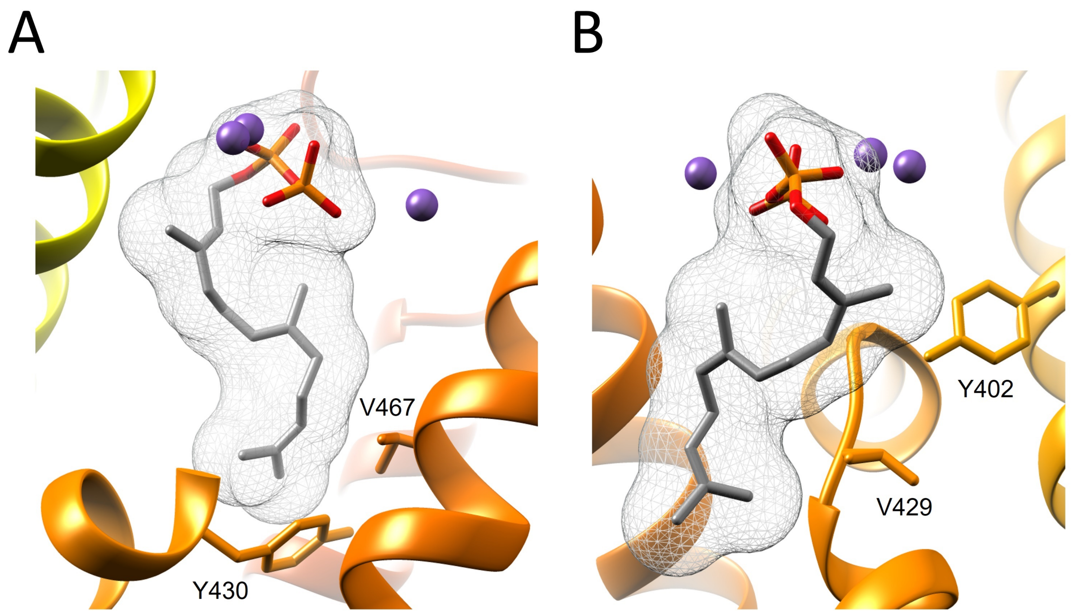


Figure 3.24: **Structural basis of epistasis on free energy landscapes.** Shown are ribbon diagram cut-outs of the BFS structural homology model (created in I-TASSER ([26], [27]) with docked FPP substrate (mesh) ([6]). Magnesium ions (purple spheres) coordinate the pyrophosphate moiety at the top of the active site. (A) Amino acids at positions 430 and 467 interact with each other and with the isopropenyl tail of the substrate at the bottom of the active site. (B) Amino acids at positions 402 and 429 are in spatial proximity with each other and with the isopropenyl tail of the substrate.

two residues, at least in the wild-type.

3.5 Fitness models

We have developed a class of biophysical fitness landscapes based on the Michaelis-Menten theory of enzyme kinetics and energetics. We start with an assumption that all products produced by a given enzyme can be classified into correct and incorrect. Making a correct product results in a fitness gain while making an incorrect product results in a fitness loss due to the necessity of its removal or degradation. In a biotechnology setting, products are classified into correct and incorrect by the researcher in a context of a specific project, whereas in a cellular setting the needs of the cell and the associated fitness gains and losses may be time- or environment-dependent. Thus, in general a single enzyme's fitness F is given by a weighted sum over fitness gains and losses associated with each product:

$$F = \sum_{i \in correct} \alpha_i n_i - \sum_{i \in incorrect} \beta_i n_i, \tag{3.9}$$

where n_i is the number of product molecules of type i produced per unit time and α_i, β_i are the corresponding fitness gains and losses per product molecule. Note that $\alpha_i = \alpha_i^{'} - \gamma$, $\beta_i = \beta_i^{'} + \gamma$, where $\gamma > 0$ is the fitness cost of making or acquiring a substrate molecule (for example, γ is expected to be close to 0 if the substrate molecules are abundant in the environment). We assume that $\alpha_i > 0, \forall i$ or, in other words, that benefits of making the correct products outweigh all the associated costs (products that are less valuable than substrates can be accounted for in the second term on the right-hand side). In the absence of information on product-specific rewards and penalties, we set all fitness gains and losses to be product-independent, which makes the fitness a weighted difference between the total number of correct and incorrect product molecules produced per unit time:

$$F = \alpha \sum_{i \in correct} n_i - \beta \sum_{i \in incorrect} n_i. \tag{3.10}$$

Note that in a cellular setting, fitness gains and losses may depend on the number of produced molecules: $\alpha_i = \alpha_i(n_i), \beta_i = \beta_i(n_i)$. For example, fitness may be maximized only if a given molecule's production rate is close to optimal; overproduction may lead to diminished returns and even sign reversal. Although such extensions are easy to model within our framework, here we focus on the product type- and product rate-independent scenario (Eq. (3.10)). For simplicity, we label all cyclic products as correct and all non-cyclic products as incorrect; alternative scenarios such as a single correct product (favoring specific enzymes typically found in nature) can be easily considered.

Within the Michaelis-Menten framework, n_i is given by the reaction velocity per enzyme molecule:

$$n_i = k_{cat,i} \frac{c}{K_{M,i} + c}, \tag{3.11}$$

where $K_{M,i}$ is the Michaelis constant and c is the substrate concentration. Thus,

$$F(c) = \alpha \sum_{i \in cyclic} k_{cat,i} \frac{c}{K_{M,i} + c} - \beta \sum_{i \in non-cyclic} k_{cat,i} \frac{c}{K_{M,i} + c}. \tag{3.12}$$

In the high substrate-concentration limit ($c \gg K_{M,i}, \forall i$), the enzyme velocity reaches its maximum value and the expression for fitness becomes

$$F(c) = \alpha \sum_{i \in cyclic} k_{cat,i} - \beta \sum_{i \in non-cyclic} k_{cat,i}. \tag{3.13}$$

In this limit, fitness is simply a function of the enzyme's reaction rates and is independent of substrate concentration. In the low substrate-concentration limit ($K_{M,i} \gg c, \forall i$),

$$F(c) = c[\alpha \sum_{i \in cyclic} \frac{k_{cat,i}}{K_{M,i}} - \beta \sum_{i \in non-cyclic} \frac{k_{cat,i}}{K_{M,i}}]. \tag{3.14}$$

In this case, fitness is proportional to the substrate concentration c and depends on both reaction rates and Michaelis constants. Moreover, if the height of all the G_3-$G_{4,i}$ barriers is low, such that $k_{cat,i} \gg k_{-1}, \forall i$, fitness becomes approximately independent of the reaction rates and the product type: $\frac{k_{cat,i}}{K_{M,i}} \approx k_1$, and the overall enzyme velocity is determined by the height of the G_1-G_2 free energy barrier (Fig. 3.1).

Note that if we do not know anything about substrate concentration *a priori*, we can assume that it is uniformly distributed in the $[c_{\min}, c_{\max}]$ range. Then the expected value of F is given by

$$\overline{F} = \frac{1}{c_{\max} - c_{\min}} \int_{c_{\min}}^{c_{\max}} dc F(c). \tag{3.15}$$

If $c_{\max} \gg K_{M,i}, \forall i$, the integral in Eq. (3.15) is dominated by the high substrate-concentration limit and we again recover Eq. (3.13). Therefore, we focus on the high substrate-concentration case (Eq. (3.13)) in the subsequent analysis.

3.5.1 Structure of the fitness landscape and epistatic interactions.

We have computed fitness values for each sequence in the combined library using Eq. (3.13). Since the overall scale of each term and therefore the absolute magnitude of the

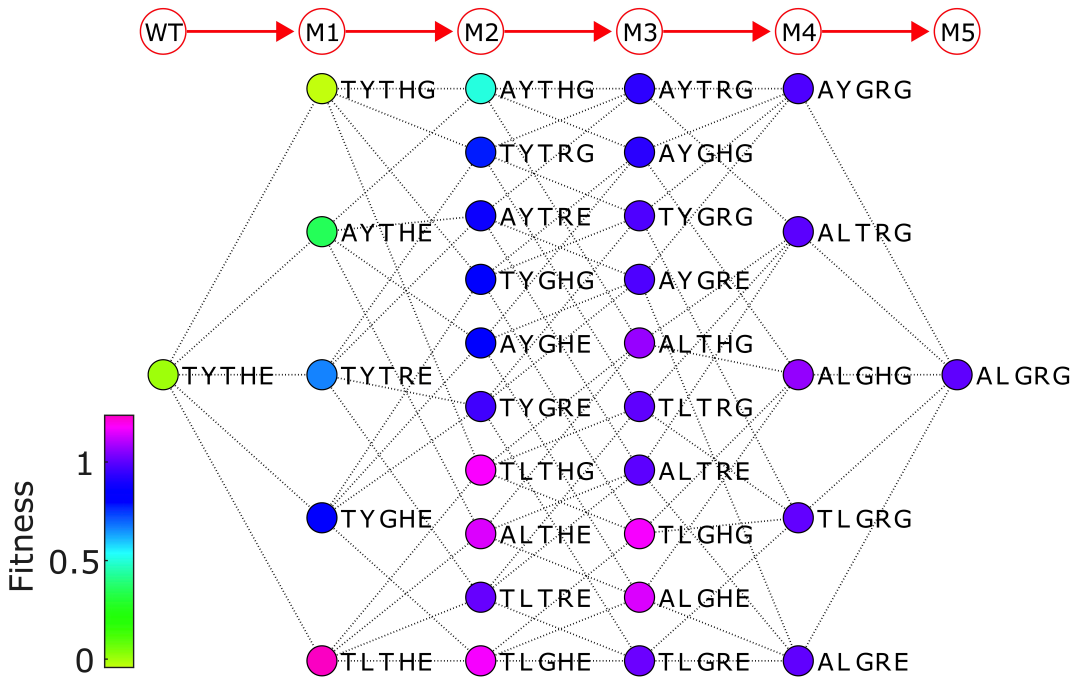


Figure 3.25: Fitness landscape for the M5 library. All fitness values were computed using Eq. (3.13) with $\alpha = 1$, $\beta = 1$ and predicted (rather than observed) reaction rates $k_{cat,i}$. The landscape is presented as in Fig. 3.10, with the nodes sorted in the same order to facilitate visual comparisons. Each node is colored according to its fitness value.

fitness contribution cannot be determined from our analysis alone, we have set $\alpha = 1$, $\beta = 1$ and have all shifted fitness values such that the fitness of the wild-type BFS sequence, TYTHE, is exactly zero (see Fig. 3.25 for the fitness landscape on the M5 library). We have chosen to use predicted rather than observed reaction rates in Eq. (3.13); switching to the experimentally observed rates would have made little difference since our model predicts reaction rates $k_{cat,i}$ with high accuracy (Fig. 3.5).

Similar to the free energy landscapes discussed above, the fitness landscape exhibits simple structure: creation of cyclic products is enabled by a single gateway mutation, Y402L, so that a single-point mutant, TLTHE, exhibits a significant jump in fitness. In fact, sequences with L at position 402 and G or T at position 429 are the best producers of cyclic products; however, the 5-point mutant, ALGRG, has somewhat lower fitness, largely because its overall reaction rate is lower (Fig. 3.13). The relatively simple structure of the fitness landscape is somewhat expected since the reaction rates in Eq.

(3.13) depend on the G_3 and G_4 free energy values, which are determined by just a few non-zero two-aa terms. On the other hand, fitness is a non-linear function of the free energies, which can lead to epistasis even if the underlying free energy model has no two-aa terms at all ([28], [29]).

To study the amount of epistasis on our fitness landscape, we have computed epistatic scores ES_{ij} using Eq. (3.8) for the 16 pairs of positions identified earlier in the epistatic analysis of free energy landscapes (Fig. 3.26). The only difference with the previous analysis is that fitness values rather than free energy values were averaged in each T_{ij} subset. We have categorized all pairs with $ES_{ij} < 0.05$ as exhibiting no epistasis. Out of the remaining 9 pairs, 8 show sign epistasis and 1 exhibits reciprocal sign epistasis, indicating that the fitness landscape is indeed rougher than the free energy landscapes considered earlier and, as a consequence, single-point mutation evolutionary trajectories are expected to be constrained.We do not observe a prominent correlation between fitness epistatic scores and the corresponding $C_\alpha - C_\alpha$ distances (Fig. 3.26): although the 430-467 and 402-429 pairs ranked first and third by the absolute magnitude of the epistatic score are separated by less than the average distance between all aa pairs, the residues in the 402-555 pair, which is ranked second, are nearly 25 apart. Strikingly, the 402-555 pair does not contribute to epistasis on any of the free energy landscapes (Fig. 3.23). Thus, considerable long-range couplings can be created purely through non-linearities in the Michaelis-Menten fitness function. In contrast, the 430-467 pair is the most significant contributor to epistatic interactions on the free energy landscapes and its residues are in direct contact with one another (Fig. 3.23, Fig. 3.24A).

Similar to the 403-467 pair, residues 402 and 429 are within 5 of one another and make direct contacts with the FPP substrate (Fig. 3.24B). Given their shared role in forming one side of the active site, the residues in the 402-429 pair are positioned to influence substrate folding and guide product formation. The role of residue 402 in activating cyclization stems from its interaction with the first isoprene unit of the substrate, which enables an initial isomerization reaction. Residue 429 resides deeper in the binding pocket and therefore more likely influences substrate folding. Together,

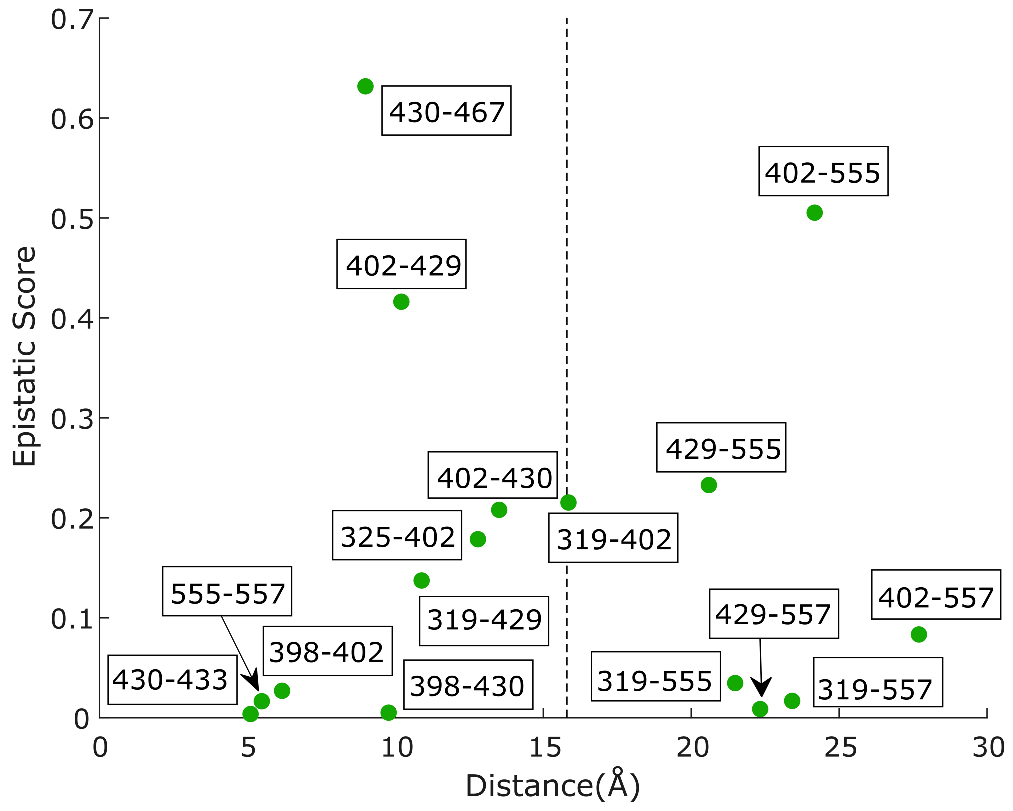
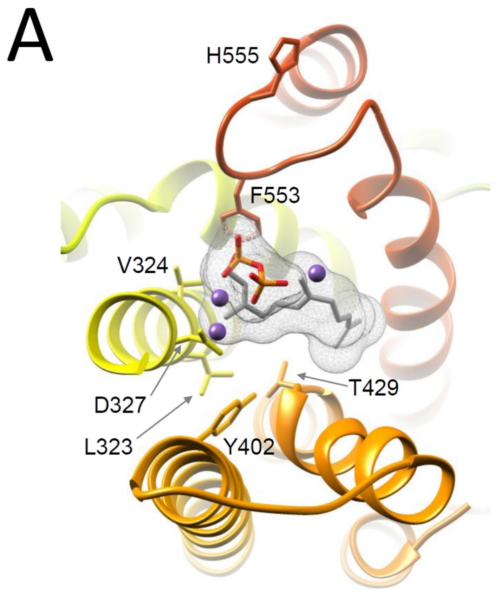


Figure 3.26: Plot of the fitness epistasis scores ES_{ij} vs. the $C_\alpha - C_\alpha$ spatial distances (in) between aa positions i and j. A horizontal solid black line indicates the 0.05 cutoff separating residue pairs considered non-epistatic from the rest. A vertical dashed grey line at 15.8 shows the average $C_\alpha - C_\alpha$ distance for all 138 aa pairs considered.

the 402-429 pair enables cyclization of a substrate conformation that readily undergoes 1,6 cyclization while the reaction is terminated by water capture, likely associated with magnesium ions positioned near the mouth of the active site. In sum, the fact that the 402-429 pair shows strong epistasis in cyclic product formation is entirely consistent with its structural role in the active site.

Given the large distance between residues at positions 402 and 555 ($\approx$ 25), we thought to rationalize the potential physical basis for interactions in the 402-555 pair through a network analysis of the BFS protein structure, whose purpose is to delineate the intervening interactions (Fig. 3.27). The network analysis identified 3 shortest pathways with 4 edges, likely not mutually exclusive, that connect residues 402 and 555 (Fig. 3.27B). Pathway 1 involves interactions exclusively between aa in the protein structure, while pathways 2 and 3 depend on intervening connections through the isopropenyl chain of the FPP substrate. Interestingly, pathway 2 transits through 429, the gateway residue that controls cyclic product synthesis. Pathway 3 transits through position 327, a conserved catalytic aspartic acid of the DDxxD motif. Subtle misalignment of the pyrophosphate-magnesium complex coordinated by the DDxxD motif, propagated through this interaction network, may provide an explanation for reduced catalytic efficiency observed upon aa substitution at position 555.



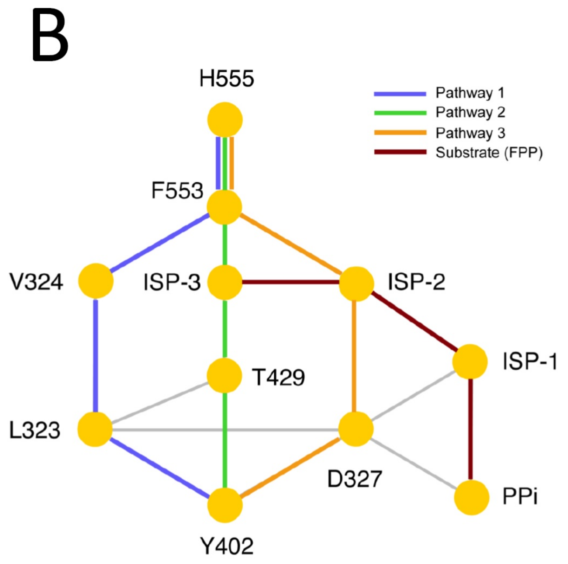


Figure 3.27: **Network analysis of structurally distant interactions in the 402-555 pair**. (A) Residues 402 and 555 are highlighted on the BFS structural homology model (created in I-TASSER ([26], [27]) with docked FPP substrate (mesh) ([6]). Residues involved in interaction networks are labeled. Magnesium ions are shown as purple spheres. (B) Shortest interactions paths between residues 402 and 555 were analyzed in Cytoscape ([30]). A network model was constructed from the BFS structural homology model using RINerator ([31]). Three shortest interaction paths of identical length were found, as indicated by color. Network edges that are not part of the shortest paths are shown in light grey. Nodes are labeled by aa type and position, by the substrate isoprene unit number (ISP-1,2,3), or as the pyrophosphate moiety (PPi). Contacts with magnesium ions are not shown.

Chapter 4

Discussion and Conclusion

We have described each enzyme in our library using the Michaelis-Menten model of enzyme kinetics ([12], [13]). The Michaelis-Menten framework allows us express enzymatic reaction rates and the overall reaction velocity in terms of free energies assigned to various enzymatic states (Fig. 3.1). These free energies are closely related to the free energies of protein folding and binding which have been extensively explored using protein engineering methods ([32], [33], [34]), with $\Delta\Delta G$ data available for multiple proteins ([35]). These studies reveal that effects of multiple mutations on protein energetics are nearly additive, especially if the mutations are distant from each other in the linear sequence ([36]). Consequently, the assumption of independent energetic contributions of residues at different sites has been extensively used in biophysical models of protein evolution that express organismal fitness in terms of protein energetics ([37], [38], [39],[29], [40]). In the light of these previous findings, we expected Michaelis-Menten free energies to be approximately additive as well, with two-aa coupling terms playing a secondary role. To check this hypothesis, we have represented the free energies as a sum of one- and two-aa contributions which we treated as fitting parameters. The resulting model, supplemented with a LASSO constraint which is designed to minimize the number of non-zero fitting parameters ([21]), was fit to the reaction rate data, reproducing it with a high degree of accuracy (Fig. 3.5).

As expected, the model has very few non-zero two-aa terms: on average, just 4.5 out of 138 coupling parameters contribute to a given free energy landscape. For example, the α-bisabolol G_4 landscape is controlled by 3 coupling parameters (Fig. 3.9). Interestingly, the (E)-β-farnesene G_4 landscape is by far the most non-additive as it is characterized by 32 non-zero coupling terms, followed by the G_3 landscape with 6 couplings (Fig. 3.7A). Since both of these landscapes affect reaction rates of the linear product of BFS, a wild-type enzyme, it is conceivable that the corresponding free energies have been evolutionarily fine-tuned, resulting in a more coupled, non-linear landscape. This implies that novel enzymatic functions can evolve quickly through pathways that do not require establishment of intricate networks of aa interactions right away. However, these networks start to play an increasingly prominent role as the novel enzyme is further optimized by evolution for efficiency and specificity.

We have sought to quantify the extent of ruggedness on the Michaelis-Menten free energy landscapes by considering epistatic interactions between various aa pairs. The notion of epistasis, including higher-order epistasis, has been extensively studied in the context of fitness landscapes ([41], [16], [42], [43]). Epistasis can profoundly alter evolutionary dynamics on fitness landscapes by restricting the availability of evolutionary trajectories ([22]) and by creating local maxima that can trap or slow down evolving populations ([25]). We have extended the idea of epistasis to the free energy landscapes; as with fitness, aa pairs have been classified into no-epistasis, magnitude epistasis, sign epistasis, and reciprocal sign epistasis categories (Fig. 3.21) ([24]). We have also introduced an epistatic score ES_{ij} which is zero in the case of no epistasis and positive otherwise (Eq. (3.8)). Note that in general this score depends not only on the magnitude of the two-aa term between residues i and j, but also on the coupling to the residues outside of the i, j pair. Consistent with the previous observations in the literature described above and the simple, easily interpretable structure of the free energy landscapes constructed in this work , we find free-energy epistasis to be of relatively limited importance, with only 15 out of 192 pairs considered exhibiting any epistasis at all, 13 of which in the magnitude epistasis category. These findings appear to be at variance with a recent report in which significant epistasis was observed in antibody-antigen binding free energies ([44]). However, we note that our analysis is effectively limited to the 2-letter alphabet and therefore our observations may change once the full spectrum of mutations is included. Furthermore, enzyme evolution, and specialized metabolic enzymes in particular, may be subject to different constraints than evolution in the adaptive immune system. Finally, our analysis is likely affected by the fact that, by design, we explore sequence space around a naturally occurring enzyme, BFS. Thus, our findings reflect ”micro” rather than ”macro” evolution which can be studied e.g. on the basis of protein sequence alignments involving multiple protein families and multiple organisms. Analysis of such alignments tends to yield much less interpretable models characterized by numerous non-zero coupling terms ([16], [17], [18], [45], [46]; [47], [48]). Interestingly, although there is a certain degree of enrichment for spatial proximity in strongly epistatic pairs, many of such pairs are separated by > 15 Å (Fig. 3.23), conceivably as a result of long-range allosteric interactions mediated by networks of intervening amino acids.

Even if the underlying free energy model is purely additive, the corresponding biophysical fitness function may be characterized by epistasis and local maxima if it is a non-linear function of the free energies, as observed in models that include protein folding stability and binding affinity as explicit fitness determinants ([37], [29]). We have significantly extended this prior work by constructing a biophysical fitness landscape in terms of free energies of the Michaelis-Menten model. Each enzyme's fitness is assumed to be proportional to its total reaction velocity for the ‘correct’ product(s) as dictated by a given biological or biotechnological context. Production of incorrect products is penalized in a similar way. Thus, highly tuned enzymes that produce the maximum number of correct molecules and the minimum number of incorrect molecules per unit time would be characterized by high fitness values. Extensions to fitness functions in which e.g. the optimal rates of production of correct products are enforced are straightforward but are not the main focus here.

As expected, the fitness landscape is more epistatic than the free energy landscapes due to its non-linear nature, with 9 out of 16 aa pairs under consideration characterized by significant epistatic scores (8 of these are in the sign epistasis and 1 in the reciprocal sign epistasis category). Thus, the fitness landscape is rougher than its underlying free energy components, which, depending on the balance between selection, mutation, and genetic drift in evolving populations, may render some of the evolutionary pathways inaccessible. Nonetheless, projecting the fitness function onto the M5 library results in a global maximum, TLTHE, and no local maxima (Fig. 3.25). Weak correlation between epistatic scores and spatial distances (Fig. 3.26) is less surprising in this case since fitness values may depend on the states of residues that are not necessarily energetically coupled, due to compensatory mutations, as for example occurs in biophysical fitness models that depend on the total free energy of protein folding and binding ([37], [49], [29]).

In conclusion, we have constructed Michaelis-Menten free energy landscapes for each enzyme of the library, study their structure quantitatively, and use them as input to a simple biophysical model of enzyme fitness. Our analysis highlights a fundamental evolutionary mechanism of creating epistatic interactions through non-linearities in the fitness function and underscores the surprising simplicity and interpretability of enzyme energetics. In the future, we intend to investigate the universality of our findings by employing additional synthetic libraries (in particular, going beyond the 2-letter alphabet), and by exploring sequence space around wild-type enzymes from other protein families.

Chapter 5

Introduction to Community Detection

Many complex systems in real life such as social media, web linkage and biological interactions can be represented as a network. All these networks contain subgroups of nodes that more densely connected to themselves than the rest of the network. These sets of nodes are called communities. Detecting these communities gives us important information about the network. For example, in Wikipedia network a community represents a set of pages of similar topics; in gene co-expression network a community represents a set of genes which have the same functionality; in social network a community represents a community of people.

Despite the intuition behind the concept, a clear definition of communities does not exist. The most strict definition of a community is a clique but it is too strict to include communities in most of the real world networks. The most widely accepted quantitative definition of a community structure in a network is the modularity score ([50]) defined by Newman as

$$M = \sum_{i} \left(e_{ii} - a_i^2 \right), \tag{5.1}$$

where e_{ii} is the fraction of edges that fall within community i and a_i is the fraction of all ends of edges that are attached to vertices in community i.

Optimizing the modularity score is the basis behind several methods. Multilevel method ([51]) greedily clusters a network in several rounds to optimizing modularity score. Leading eigenvector ([52]) method uses the largest eigenvector of the modularity matrix to cluster a network. Algorithms such as Walktrap ([53]) use random-walks to extract tight-knit communities while Label Propagation ([54]) uses labels randomly propagated through the network to identify communities.

In our method, we assume that a community is a set of nodes such that random-walks restricted in a community reach steady state in a few steps. Based on this assumption we calculate properties of random-walks on community structured networks. These properties are exploited to find the marginal probability of community identity of each node of the network, hence partitioning the network into communities. On testing our algorithm on multiple artificial and real-world networks, we find that it performs better or as good as other competitive algorithms.

Chapter 6 talks about properties of random-walks on complex and community structured networks. This formalism is used in proposing a community detection algorithm in Chapter 7. We validate the performance of our algorithm on several artificial and real-world networks in Chapter 8. Finally, Chapter 9 talks about the conclusion and other ideas derived from this project.

The main motivation behind this particular method for community detection came through our former project aimed at finding community statistics of a network by random-walks, mentioned in the appendix of the thesis. This was a continuation to a previous work of the group ([55]). The project was aimed at finding statistics of network community structure such as the number of communities and the size of each community and asymptotically cluster the whole network into communities. Although this method of community detection was successful in its aim, it was neither competitive in speed nor in accuracy to other community detection methods, making the method redundant. Despite these severe shortcomings, the method provided all the necessary tools and ideas for development of the current method for community detection.

Chapter 6

Random-Walks on complex networks

Consider a network with symmetric edge weights: $w_{ij} = w_{ji}$ where w_{ij} is weight of edge joining node j to node i (Fig. 6.1). Consider discreet random-walks on this network such that the probability of transition from node i to node j in one step is given by

$$P(i \to j) = \frac{w_{ij}}{\sum_{k \in nn(i)} w_{ik}}, \tag{6.1}$$

where $nn(i)$ refers to the nearest neighbors of the node i. In this type of random-walks the steady state probability of the node i ($P_{st}(i)$) is proportional to the sum of weights of the edges (w_i) contained by the node

$$P_{st}(i) = \frac{w_i}{\sum_{j \in \text{Network}} w_j}, \tag{6.2}$$

where $w_i = \sum_j w_{ij}$.

Now, consider a continuous time random-walk on the same network. At each quanta of time δt, the random-walker leaves the node on which it is currently with probability

$$P = \delta t / \tau \tag{6.3}$$

where τ is the mean waiting time on each node. Thus, the probability of staying on the same node for time t is $e^{-t/\tau}$. Similarly the probability of making ℓ^{th} jump at time t is given by

$$P(k,t) = \frac{1}{(\ell-1)!}(\ell-1)^t e^{-t/\tau} \tag{6.4}$$

After the random-walker jumps away from node i, it goes to j with the same probability as in discreet time random-walks. Many properties of this type of random-walks, like

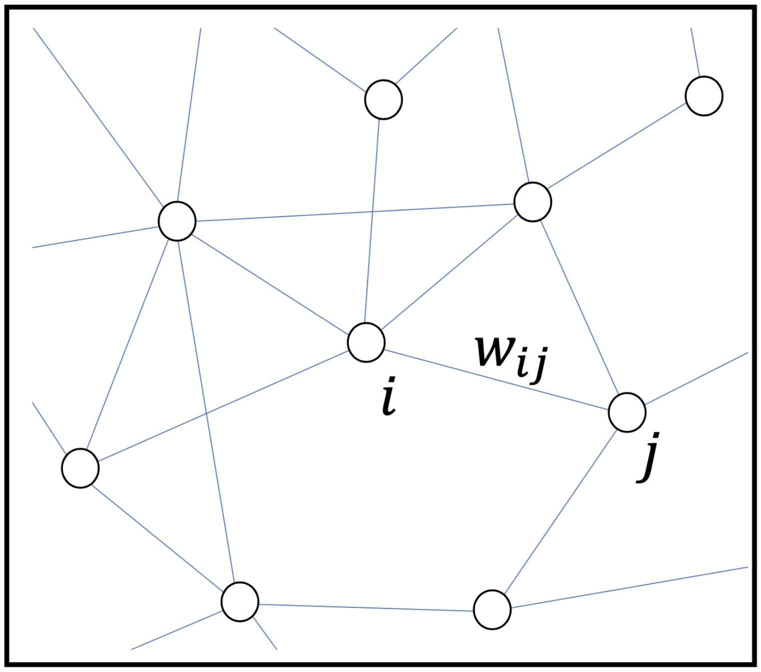


Figure 6.1: Random-walks on a complex network

the steady state probability are the same as that of the discreet random-walks. We shall use the continuous-time random-walks to calculate some first-passage properties and then convert these values for discreet time random-walks.

6.1 Mean Return Time

Consider a continuous time random walk with mean waiting time $\tau = 1$. Now, if the random-walker was on node n at time $t = 0$ and is at the same node after time t, the possibilities are that the random-walker did not jump till time t, or stayed there for some time $t' < t$ and returned in rest of the time. Thus for $P(n,t)$: probability that random-walker is at the same node n at time t and $F(n,t)$: probability to return to the node n for the first time at time t

$$P(n,t) = e^{-t} + \int_0^t e^{-t'} F(n, t-t')dt' + \int_0^t \int_0^{t_2} e^{-t_1} F(n, t_2 - t_1) F(n, t-t_2) dt_1 dt_2 + ... \tag{6.5}$$

Taking Laplace transform we get

$$\tilde{P}(n,s) = \frac{1}{1+s}\left(\sum_{i=0}^{\infty}(\tilde{F}(n,s))^i\right) = \frac{1}{(1+s)(1-\tilde{F}(n,s))}, \tag{6.6}$$

where $\tilde{P}(n,s) = \int_0^\infty P(n,t)e^{-st}dt$ and $\tilde{F}(s) = \int_0^\infty F(t)e^{-st}dt$. Thus,

$$\tilde{F}(n,s) = 1 - \frac{1}{(1+s)\tilde{P}(n,s)} \tag{6.7}$$

Let $S_0(n,t) = 1 - \int_0^t F(n,t')dt'$ be survival probability. Thus,

$$\tilde{S_0}(n,s) = \frac{1}{s}(1-\tilde{F}(n,s)) = \frac{1}{s(1+s)(\tilde{P}(n,s))} \tag{6.8}$$

Let $H(n,s) = \int_0^\infty (P(n,t) - P_{st}(n))e^{-st}dt$ where $P_{st}(n)$ is the steady state probability of the node n. Note that $H(n,0)$ is finite but $\tilde{P}(n,0)$ is infinite. So, our equation becomes

$$\tilde{S_0}(n,s) = \frac{1}{(1+s)(P_{st}(n) + sH(n,s))} \tag{6.9}$$

But the mean return time $\langle T(n)\rangle$ to the node n is $\int_0^\infty tF(t)dt = \int_0^\infty S(t)dt$. Thus

$$\langle T(n)\rangle = \lim_{s\to 0}\tilde{S_0}(n,s) = \frac{1}{P_{st}(n)} = \frac{\mathcal{W}}{w_n}, \tag{6.10}$$

where $\mathcal{W} = \sum_{n\in\text{network}} w_n$.

Now, let $F_c(n,t)$ refer to the return time probability for continuous time random-walks and $F_d(n,\ell)$ refer to the probability to return to the node n for the first time at ℓ^{th} step in discreet time random-walks. From equation 6.4 we get

$$F_c(t) = \sum_{\ell=0}^{\infty}\frac{1}{(\ell-1)!}(\ell-1)^t e^{-t}F_d(\ell) \tag{6.11}$$

But since $\langle T(n)\rangle_d = \sum_{\ell=0}^\infty \ell F_d(n,\ell)$ and $\langle T(n)\rangle_c = \int_0^\infty tF_c(n,t)dt$ we get

$$\langle T(n)\rangle_c = \int_0^\infty \sum_{\ell=0}^{\infty}\frac{(\ell-1)^t}{(\ell-1)!}e^{-t}F_d(\ell)dt = \sum_{\ell=0}^{\infty}\ell F_d(n,\ell) = \langle T(n)\rangle_d \tag{6.12}$$

Thus, the mean return time is exactly same in continuous and discreet time random-walks.

6.2 Mean Return Time to a multiple nodes

Consider random-walks starting from a set of nodes $S = \{n_1, n_2, ..., n_{N_p}\}$. The initial condition of the random-walks is

$$P(n_i, t = 0) = \frac{w_{n_i}}{\mathcal{W}_S}. \tag{6.13}$$

where $\mathcal{W}_S = \sum_{n_i \in S} w_{n_i}$. To find the return time to this set, we tweak the network by replacing all the nodes of the set S by a single node n_s as shown in Fig 6.2. Thus, the new network has a total of $N - N_p + 1$ where N is the number of nodes in the original network. The new network contains edges connecting node i to node j with edge weights w'_{ij} in the following way

- $w'_{ij} = w_{ij}$ for $i, j \notin S$
- $w'_{is} = \sum_{j \in S} w_{ij}$ for $i \notin S$
- $w'_{ss} = \sum_{i \in S} \sum_{j \in S} w_{ij}$

Thus, for the node n_s, $w'_{n_s} = \sum_{n_i \in S} w_{n_i}$. This keep the sum of the weights of all the links in the new network same as before, resulting in

$$P'_{st}(s) = \sum_{n_i \in S} P_{st}(n_i) \tag{6.14}$$

where $P'_{st}(n_s)$ refers to the steady state probability of node n_s in the new network. Further, the probability to step away from the node n_s is same as the probability to step away from all the nodes of the set S and so is the random-walks on rest of the network without visiting the set of the nodes S. Thus, the return time process on the set of nodes S is exactly same as that in this modified node n_s. Hence,

$$\langle T(S) \rangle = \frac{\mathcal{W}}{\mathcal{W}_S}, \tag{6.15}$$

6.3 Higher moments and Return Time distribution

In this subsection we attempt to make approximations to the return time distribution by finding higher order moments of return time. Let, $S_1(n, t) = \int_t^\infty S_0(n, t')dt'$. Thus,

$$\tilde{S}_1(n, s) = \frac{1}{s}(\tilde{S}_0(n, 0) - \tilde{S}_0(n, s)) = \frac{H(n, s) + P_{st}(n) + sH(n, s)}{(1 + s)(P_{st}(n) + sH(n, s))P_{st}(n)} \tag{6.16}$$

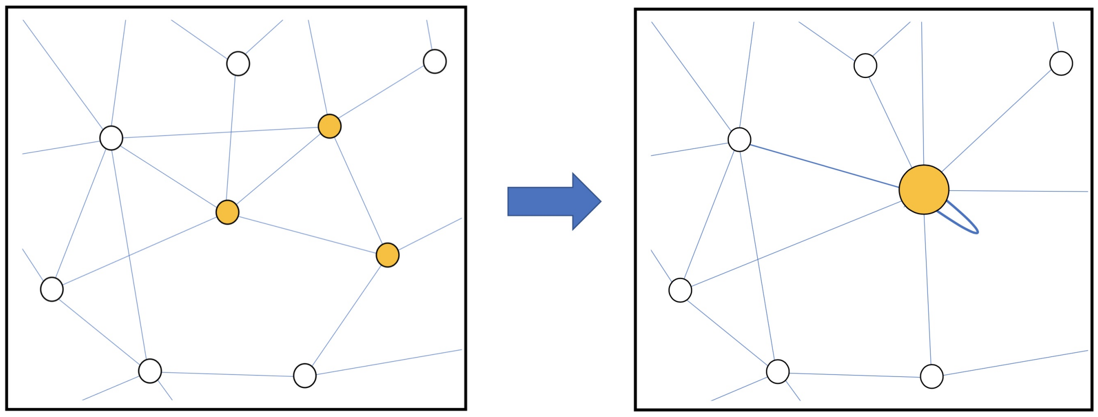


Figure 6.2: Modifying network by merging multiple nodes

Thus,

$$\langle T^2(n)\rangle = 2!\tilde{S}_1(n,0) = 2\left(\frac{1}{P_{st}(n)}\right)^2 (H(n,0) + P_{st}(n)) \tag{6.17}$$

Using similar approach, in the assumption $N >> 0$ we get,

$$\langle T^k\rangle \approx k!\left(\frac{1}{P_{st}}\right)^k (H(n,0) + P_{st}(n))^{k-1} \tag{6.18}$$

Thus, we can approximate the survival probability as

$$S(n,t) = \frac{1}{G(n)} e^{-\frac{P_{st}(n)}{G(n)}t}, \tag{6.19}$$

for $G(n) = \int_0^\infty (P(n,t) - P_{st}(n))dt + P_{st}(n)$.

Converting from continuous time to discreet random walks we get

$$\langle T^k(n)\rangle_c = \sum_{\ell=0}^{\infty} \frac{(\ell+k-1)!}{(k-1)!} F_d(n,\ell) \tag{6.20}$$

Assuming that $N >> k$, we get $\langle T(n)^k\rangle_c \approx \langle T(n)^k\rangle_d$. Thus, equation 6.19 can be used for discreet random-walks as well.

6.4 Random-walks on community-structured networks

Note that the quantity $G(n)$ in equation 6.19 is always greater than 1. The more connected the network is the closer it is to 1. We assume that a community is such a highly connected subunit of a network such that if the community C is disconnected from the rest of the network, $G(S) \approx 1$ if $S \in C$. Thus, the probability to return to the set S for the first time after ℓ steps, given that the random-walks is restricted to the community is

$$P(\ell) = \frac{N_p\langle w\rangle_p}{N_C\langle w\rangle_C} e^{-(N_p\langle w\rangle_p/N_C\langle w\rangle_C)\ell} \tag{6.21}$$

where N_p is the number of nodes in the set S, $\langle w\rangle_p$ is their average connectivity, N_C is the number of nodes in community C and $\langle w\rangle_C$ is their average connectivity. We shall refer to the sum of connectivity of a set of nodes as the weighted size of the nodes denoted by $\mathcal{W}_p = N_p\langle w\rangle_p$ and $\mathcal{W}_C = N_C\langle w\rangle_C$. It then follows that the probability to not make a return for ℓ steps (i.e. the survival probability) given no transition outside the community

$$S(\ell) = e^{-(\mathcal{W}_p/\mathcal{W}_C)\ell} \tag{6.22}$$

Thus, the probability of making $\mathcal{K}$ returns to the nodes of set S in ℓ_C steps in the community C is given by a poisson distribution

$$P(\mathcal{K}|\mathcal{W}_C, \ell_C) = \text{Pois}\left(\mathcal{K}, \frac{\mathcal{W}_p\ell_C}{\mathcal{W}_C}\right) = \frac{1}{\mathcal{K}!}\left(\frac{\mathcal{W}_p}{\mathcal{W}_C}\ell_C\right)^K e^{-(\mathcal{W}_p/\mathcal{W}_c)\ell_C} \tag{6.23}$$

Given $\mathcal{K}$ and $\mathcal{W}_p$, and assuming uniform prior the we can obtain an estimate of the quantity $\mathcal{X}_C = \ell_C/\mathcal{W}_C$

$$\hat{\mathcal{X}}_C = \frac{\mathcal{K}}{\mathcal{W}_p} \tag{6.24}$$

The quantity $\mathcal{X}_C$ can be used in finding community identity of unknown nodes in the following way. Consider the events where a node i is visited k_j times in random-walks $\{RW_j\}_{j=1}^n$ on a network with m communities $\{C_s\}_{s=1}^m$. Assume that in the random-walks RW_j, the random-walker takes $\ell_{jC'}$ on each community C' of weighted size $\mathcal{W}_{C'}$. Now, if the node $i \in C$, then the probability of visiting the node i, k_j times in random-walks RW_j is given by

$$P(\{k_j\}|i \in C, \{\ell_{jC}, \mathcal{W}_C\}) = \Pi_{j=1}^n \text{Pois}(k_j, w_i\mathcal{X}_{jC}) \tag{6.25}$$

where $\mathcal{X}_{jC} = \ell_{jC}/\mathcal{W}_C$ and w_i is the connectivity of the node i. Thus, if the community of the node i is not known, using Bayes' theorem, we can invert this probability to find the probability that the node $i \in C$

$$P(i \in C | \{k_j\}, \{\mathcal{X}_{jC}\}) = \frac{\Pi_{j=1}^{n} \mathrm{Pois}(k_j, w_i \mathcal{X}_{jC})}{\sum_{\{C'\}} \left(\Pi_{j=1}^{n} \mathrm{Pois}(k_j, w_i \mathcal{X}_{jC'}) \right)} \tag{6.26}$$

Chapter 7

Community Detection Algorithm

Consider a network with N nodes $\{n_i\}_{i=1}^{N}$ and m communities $\{C_i\}_{i=1}^{m}$. Let $A_{N\times N}$ be the transition matrix of the community where $A_{n'n} = w_{n'n}/w_n$ is the probability to jump from node n to node n' in 1 step. We consider a matrix $U_{N\times m}$: the belief of the partition of the network such that each element U_{nc} is the belief that the node n belongs to the community c. Consider V such that

$$V = \sum_{l=1}^{l_{max}} A^l \tilde{U} \tag{7.1}$$

where $\tilde{U}$ is a matrix such that $\tilde{U}_{nc} = \frac{w_n U_{nc}}{\sum_{i=1}^{N} w_n U_{nc}}$. V_{nc} is the expected number of times the node n is visited in l_{max} step random-walks started from nodes of community c. Assume that there are G_c such l_{max} step random-walks which started from nodes of community c and let X_{nc} be the random variable which represents the cumulative visits to the node n in these G_c random-walks starting from community c. Note that the mean of the random variable X_{nc} is $G_c V_{nc}$. Now, if $G_c = tg_c$ then for $t \to \infty$, $P(X_{nc} = tg_c V_{nc}) \to 1$. Thus, in thelimit $t \to \infty$ we use $tg_c V_{nc}$ as an observation for the random variable X_{nc}. Assuming that the community identity of the node n is not known, we get the probability that node n belongs to community c using equation 6.26.

$$P(n \in c | \{tg_{c'} V_{nc'}\}, \{\mathcal{X}_{c'c}\}) = \lim_{t\to\infty} \frac{\Pi_{\{c'\}} \text{Pois}(tg_{c'} V_{nc'}, \mathcal{X}_{c'c} w_n)}{\sum_{\{c''\}} \left(\Pi_{\{c'\}} \text{Pois}(tg_{c'} V_{nc'}, \mathcal{X}_{c'c''} w_n)\right)} \tag{7.2}$$

where $\mathcal{X}_{c'c}$ is the number of steps of random-walks c' in community c per the weighted size of community c. Using equation 6.24, we get $\mathcal{X}_{cc'} = tg_c (V^T U)_{cc'} / \sum_{\{n\}} U_{nc'} w_n$ as $tg_c (V^T U)_{cc'}$ is the number of visits from random-walks c to marked nodes of community c' and $\sum_{\{n\}} U_{nc'} w_n$ is the weighted size of the marked nodes of each community. In the limit $t \to \infty$, equation 8 simplifies to

$$P(n \in c) = \delta_{\tilde{c}c} \text{ for } \tilde{c} = \text{argmax}_{c''} \left(\sum_{\{c'\}} (V_{nc'} \log(Q_{c'c''}) - Q_{c'c''} w_n) g_{c'} \right) \tag{7.3}$$

where $Q_{cc'} = (V^T U)_{cc'} / \sum_{\{n\}} U_{nc'} w_n$. Thus, we can assign community identities to each node n. These community identities serve as the matrix U for the next iteration ($U_{nc} = P(n \in c)$).

Choice of gauge parameters g_c: The gauge parameter g_c determines the relative weights of poisson distributions associated with random-walks which start from community c. To ensure uniformity we set g_c such that the mean visits to a node $n \in c$ from random-walks starting from community c is independent of the parameters associated with community c. Since the mean visits to the node n is $g_c Q_{cc} w_n$, we get $g_c = 1/Q_{cc}$.

Choice of l_{max}: l_{max} should be big enough so that the random-walker diffuses well in the starting community but not big enough to diffuse throughout the network. Therefore, the best choice of l_{max} is the diameter of the network.

7.1 Walk-likelihood function

Thus, using the idea mentioned above we define the walk-likelihood function which inputs U: the belief of the partition of the network and A: the transition matrix and outputs $U^{(o)}$: the optimized partition of the network that has the same number of communities as U. The algorithm is loop over the following 3 steps

1. Perform l_{max} steps random walks from each community:
   $V_{nc} \leftarrow \sum_{l=1}^{l_{max}} \left(\sum_{n'} (A^l)_{nn'} w_{n'} U_{n'c} \right)$

2. Compute the rate of transition between communities:
   $Q_{cc'} \leftarrow \frac{(V^T U)_{cc'}}{\sum_n w_n U_{nc'}}$

3. Based on marginal probabilities, for each node n reassign $U_{nc} \leftarrow \delta_{\tilde{c}_n c}$ where
   $\tilde{c}_n = \text{argmax}_{c''} \left(\sum_{c'} \frac{1}{Q_{c'c'}} (V_{nc'} \log(Q_{c'c''}) - Q_{c'c''} w_n) \right)$

Halting Criteria: To determine if the algorithm has converged we find the similarity partitions in subsequent iteration normalized mutual information. The normalized mutual information (NMI) between partitions U and U' is defined as

$$\mathrm{NMI}(U,U') = \frac{2\left(\sum_{c=1}^{m}\sum_{c'=1}^{m'} P_{UU'}(c,c')(\log P_{UU'}(c,c') - \log P_U(c)P_{U'}(c'))\right)}{\sum_{c=1}^{m} P_U(c)\log P_U + \sum_{c=1}^{m'} P_{U'}(c)\log P_{U'}(c))}, \quad (7.4)$$

where $P_U(c) = \sum_i U_{ic}/N$, $P_{UU'}(c,c') = \sum_i U_{ic}U'_{ic'}/N$, m and m' refer to the number of communities of the partition U and U' respectively, and N is the total number of nodes in the network. Note that in this case $m = m'$. NMI is always between 0 and 1. If $\mathrm{NMI}(U,U') = 1$, the partitions U and U' are exactly same. Thus, we halt the process if NMI between partitions obtained in subsequent iterations of the algorithm is greater than the given threshold.

7.2 Initializing belief

Since the algorithm clusters the data in the assumption that the properties of the belief of the partition of the network reflect the properties of the true partition of the network, we require the belief of the network to be at least partially accurate. We use the following methods for initializing the belief.

7.2.1 Non-negative Matrix Factorization

Non negative matrix factorization is a method of decomposing a non-negative matrix $X_{N_1\times N_2}$ into two non-negative matrices $L_{N_1\times m}$ and $R_{m\times N_2}$ ([56],[57],[58]).

$$X = LR \quad (7.5)$$

For clustering a graph into m communities using this method, the adjacency matrix of the graph X is factorized into two non negative matrices L and R. We define U such that $U_{ic} = \mathrm{argmax}_i L_{ic}$ to obtain the partition of the graph into m communities. We use sklearn package of python for NMF.

7.2.2 Non-negative Double Single Value Decomposition

Non-negative Double Single Value Decomposition (NNDSVD) is a method to convert the singular value decomposition (SVD) of a non negative matrix X to obtain an approximate factorization X into two non negative matrices L and R ([59]). NNDSVD is typically used as an initialization for NMF. We use the same method to obtain the partition of the network from the matrix L as mentioned above for NMF. Note that the partition obtained from NNDSVD often contain nodes which haven't been assigned to any community. NNDSVD provides a fast initialization whereas NMF provides a slower but more accurate one.

7.3 Walk-likelihood bifurcation algorithm

Based on the ideas mentioned above, we define the Walk-likelihood bifurcation (WL bifurcation) algorithm to partition a network into communities when the number of communities in the network is unknown. We start this algorithm by assuming that the whole network is a single community. The algorithm is a loop over the following two steps

1. In this step, we initialize the belief for the walk-likelihood function by bifurcating each community into two more communities using either SVD or NMF. Note that at the start of the algorithm, this step bifurcates a network into two communities.

2. Applying the walk-likelihood algorithm, a more accurate division of the network is formed. In order to check if the division of the communities is optimal, for all pairs of communities, we check if combining any pair increases the modularity score of the partition ([50]). The change in modularity after merging communities c and c' is given by

$$\Delta M_{cc'} = 2(e_{cc'} - a_c a_{c'}) \tag{7.6}$$

where $e_{cc'} = (U^T AU)_{cc'}/\sum_n w_n$ and $a_c = \sum_n U_{nc} w_n / \sum_n w_n$.If there exists at least one pair of communities such that $\Delta Q_{cc'} > 0$, we merge the pair of communities c and c' for which the increase in modularity score is maximum. After merging the two communities, we run the walk likelihood algorithm again with the belief of the partition being the current partition of the network. This process of merging communities and running the walk-likelihood function is continued till there is no pair of communities left that increases the modularity score on merging.

Halting Criteria: Similar to the walk-likelihood function, we halt this process if the NMI between partitions obtained in subsequent iterations of the algorithm is greater than the given threshold and the number of communities in two partitions remain constant. The algorithm is also halted in case the number of communities decreases in subsequent iterations.

In case a community cannot be further bifurcated, we avoid the attempt to bifurcate it twice by checking if there is any match between communities of the current partition and those of the previous partition. This is done by calculating the score

$$E_{cc'} = 1 - \frac{2\sum_i U_{ic}^{(t)} U_{ic'}^{(t-1)}}{\sum_i (U_{ic}^{(t)} + U_{ic'}^{(t-1)})} \tag{7.7}$$

between the communities c and c' of current iteration ($U^{(t)}$) and previous iteration ($U^{(t-1)}$) respectively. If $E_{cc'} < \epsilon$ for the threshold ϵ, we assume that communities c and c' are the same and are not bifurcated again in step 1.

Fig 7.1 illustrates the functioning of this algorithm on a network having a true partition of 3 communities C_1, C_2 and C_3. Step 1 of the first iteration of the algorithm bifurcates the network into two communities using either NMF or SVD. Notice that division is not completely accurate. In Step 2, walk-likelihood function is used on the network which results in a more accurate division of the network into two communities. In Step 1 of iteration 2, each community is further divided into two, resulting in a total of 4 communities. In Step 2, the walk-likelihood function gives a more accurate division of the yellow and blue communities. The two communities inside C_1 in the Step 1 are merged, since the modularity of the partition increases on their merging. In Step 1 of the third iteration each community other than the red community is bifurcated. This is because there was an unsuccessful attempt to bifurcate the red community in the previous iteration. In Step 2 of iteration 3, the communities inside C_2 merge and

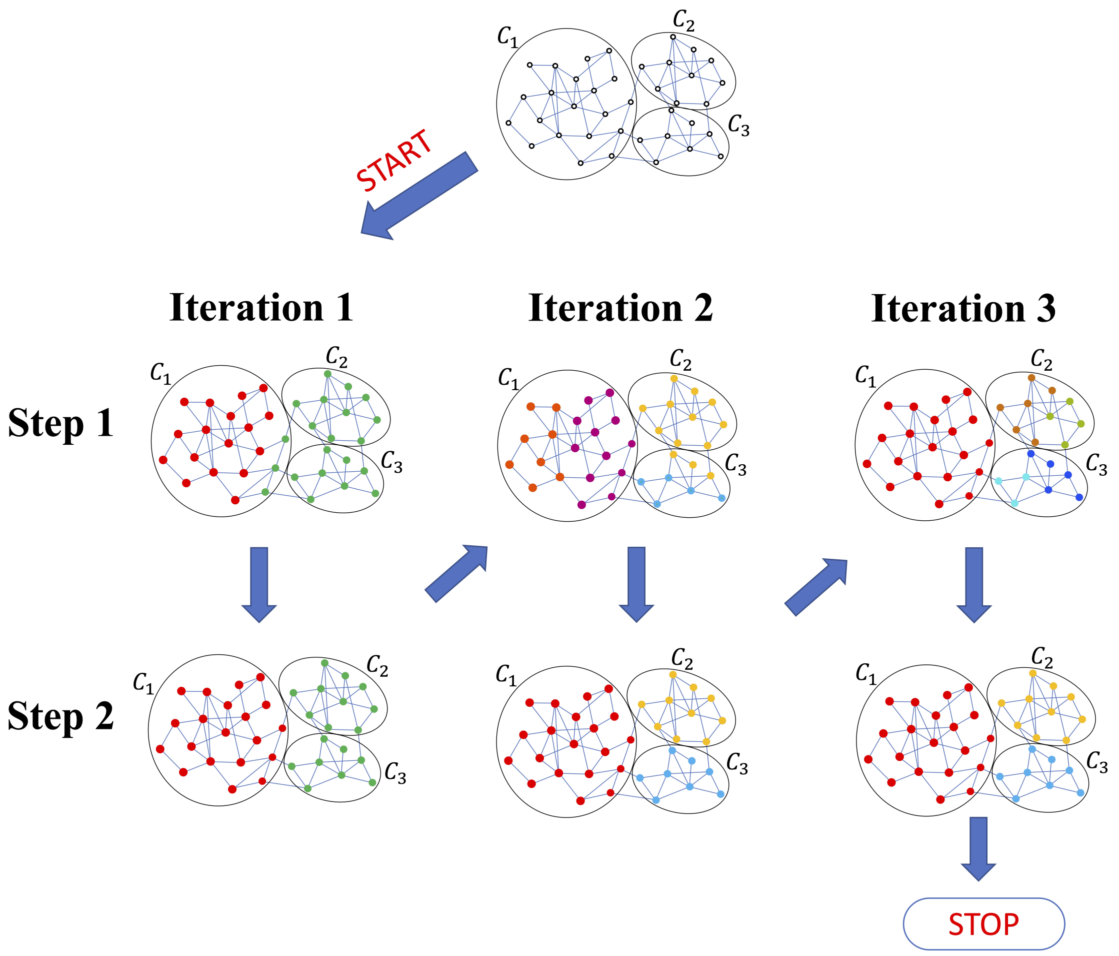


Figure 7.1: Functioning of walk-likelihood algorithm on a network with three communities

so do the communities inside C_3, resulting in the same partition at the end of the second iteration. Since, the partitions over subsequent iterations remain the same, the algorithm is halted.

Chapter 8

Experiments

We have done thorough experiments on both artificial and real world networks to check the performance of our algorithm. For each network we have compared the performance of our algorithm against 4 other community detection algorithms; multilevel, leading eigenvector, label propagation and non-negative matrix factorization.

8.1 Algorithms

8.1.1 Non-negative matrix factorization bifurcation

This is the algorithm as walk-likelihood bifurcation with NMF initialization but without using the walk-likelihood function. Thus, this algorithm bifurcates each community using NMF till no bifurcations are possible. The NMF bifurcation serves as a null model of the WL bifurcation algorithm.

8.1.2 Leading Eigenvector

The next community detection algorithm we compare to is the Leading Eigenvector proposed in [52] based on the concept of ”modularity” by the same author. First the modularity matrix of the network is constructed and the leading eigenvector computed. The network is then split into two segments based on this eigenvector such that the modularity is maximized. This process is repeated on the subdivisions of the network until no further contribution to the modularity is found. The time complexity of this algorithm is known to be $\mathcal{O}(N(E+N))$ or $\mathcal{O}(N^2)$ if the network is sparse [60].

8.1.3 Multilevel

This algorithm was originally proposed as both a simple, yet very competitive method for community detection, and is also based on the concept of network modularity [54]. The algorithm is as follows: first each node in the network is randomly assigned to a unique community. Then each node is assigned to the same community as one of its nearest neighbors, chosen such that the assignment results in the greatest increase in modularity. Nodes of the same community are then considered single nodes, and this process of reassignment is repeated until either no single node remains, or there is no reassignment which will further increase the modularity. The time complexity of such a process is necessarily $\mathcal{O}(N \log N)$ [60].

8.1.4 Label Propogation

The final algorithm we consider for comparison is that of Label Propogation ([51]). Once nodes are assigned to an initial random community, a random list of all nodes is sequentially traversed. Each node in the list is assigned to the community of the majority of its nearest neighbors. This process is repeated until no new assignments can be made. This algorithm inherently makes the assumption that each node should belong to the same community as the majority of its nearest neighbors leading to a near linear time complexity of $\mathcal{O}(E)$ per list traversal [60].

8.2 Artificial Networks

To test the performance of our algorithm in a controlled setting on a realistic synthetic network with tunable properties, we have chosen the Lancichinetti, Fortunato and Radicchi network (LFR) [61] network. The LFR benchmark network replicates some behaviour of the real world networks with a power law distribution of degree and community size with exponents β and γ respectively. The LFR networks are also characterized by the mixing parameter, μ, defined as

$$\mu = \frac{\sum_i k_i^{\text{ext}}}{\sum_i k_i^{\text{tot}}} \tag{8.1}$$

Parameter	Value
Number of nodes N	233-100,000
Maximum degree	$0.1N$
Maximum community size	$0.1N$
Average degree	20
Degree distribution exponent	-2
Community size distribution exponent	-1
Mixing coefficient μ	$0.03 - 0.75$

Table 8.1: Parameters of LFR Benchmark Networks

where k_i^{ext} is the number of links between node i and nodes in other communities, k_i^{tot} is the total links of node i, and the sum runs over all nodes in the network [60]. This parameter is defined in the interval from 0 (strong community structure) to 1 (no community structure). The closer μ is to 1, the more challenging the task of community detection becomes. We have generated networks of sizes varying from 233 to $100,000$. For each size, we have 25 different mixing parameters ranging from 0.03 to 0.75. To deal with possible discrepancies in the network properties, we have randomly generated 100 network for every set of parameters for $N < 10,000$. Due to increase in time complexity with size we generated 20 network for every set of parameters for $N > 10,000$. Other parameters used in generation of these networks are listed in Table 8.1.

Figure 8.1 shows comparison of five different methods applied to the LFR networks. As expected all the algorithms perform almost perfectly on lower values of mixing parameter μ and perform poorly on higher values of μ. The performance of the algorithms for the same value of μ also decrease as the size of the network increases. We notice that our algorithms: WL bifurcation method with NMF initialization and WL bifurcation method with SVD initialization outperform every other algorithm for LFR networks of all sizes. The difference is performance is highlighted the most for $N = 100,000$ where the NMI score of our algorithms better the second best NMI score by more than 15% for μ between 0.1 and 0.5. For networks of other sizes, the difference in performance is specially evident at higher value of mixing parameter μ.

We also notice that the NMF initialization gives a better performance for our algorithm than SVD initialization, although the difference is marginal in most cases.

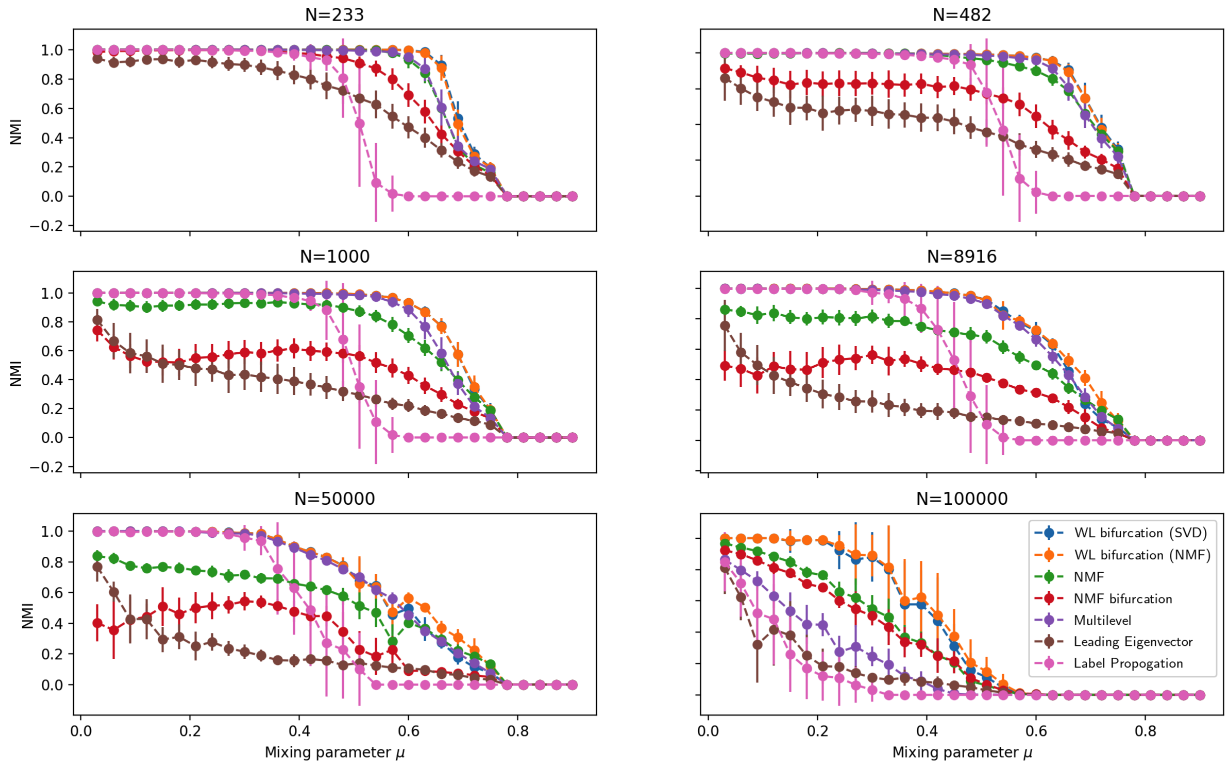


Figure 8.1: Comparison of five different community detection algorithms on the LFR benchmark for distinct values of N. The parameters for generating the LFR benchmark networks are mentioned in Table 8.1

Comparing WL bifurcation to it's null model NMF bifurcation, we notice a huge difference between their NMI values, highlighting the role of walk-likelihood function in our algorithm.

8.3 Real-World Networks

For real world networks, we have applied our algorithm to eight well known networks described below:

- **Bottlenose dolphins network:** A network of group of dolphins of Doubtful Sound, New Zealand observed by David Lusseau, a researcher at the University of Aberdeen ([62]). Every time a school of dolphins was encountered, each dolphin of the group was identified using from natural markings on the dorsal fin. This information was utilised to form a social network where each node represents dolphins and edges representing their preferred companionship.

- **Les Miserables network:** A network of co-appearance characters in the novel Les Miserables ([63]). Each node represents a character and the edge represents their co-occurrence.

- **American college football teams network:** A network of all Division IA college football games during the regular season in Fall 2000 with each node indicating a team's respective conference and the edge weight indicating the number of games between teams ([64]).

- **Jazz musicians network:** This is the collaboration network between Jazz musicians ([65]). Each node is a Jazz musician and an edge denotes that two musicians have played together in a band.

- **C. elegans neural network:** Each node represents a neuron and each edge represents its connection with other neurons ([66]).

- **Erdős coaouthorship network:** A network of Paul Erdos, his co-authors and their co-authors. Each node represents an author and there is an each edge between two authors if they have co-authored a paper ([67],[68]).

- **Edinburgh Associative Thesaurus network:** A network of word association norms showing the counts of word association as collected from British university students around 1970. Nodes are English words, and a link exists between A and B denotes if the word B was given as a response to the stimulus word A in user experiments [69].

- **High-energy physics theory citation network:** A citation network of High-energy physics theorists. Each node represents an author and there is an each edge between two authors if they have cited each other in a paper ([70]).

Network	N	$\langle w \rangle$	WL bifurcation	Multilevel	Leading Eigenvector	Label Propagation
Dolphin	62	5.13	**0.5277(5)**	0.5188(5)	0.4912(5)	0.4648(3)
Lesmis	77	6.60	0.5519(6)	**0.5555(6)**	0.5323(8)	0.5242(5)
Football	115	10.66	**0.6044(10)**	0.6043(10)	0.4926(8)	0.5804(8)
Jazz	198	27.70	0.4428(3)	**0.4447(4)**	0.3936(3)	0.2813(3)
C. elegans	297	15.80	**0.4023(5)**	0.3983(6)	0.3415(5)	0.2152(2)
Erdos02	6927	3.42	**0.6962(24)**	0.6925(31)	0.5979(27)	0.5937(326)
EAT-RS	23219	67.95	**0.4793(17)**	0.4283(4)	0.3123(4)	0.0(1)
HEP-th-new	27770	25.41	0.6524(18)	**0.6560(171)**	0.5010(152)	0.3361(498)

Table 8.2: **Comparison of the modularity and number of groups (inside parenthesis) for different community detection algorithms applied in some real-world networks.**

Results of Table 8.2 show that walk-likelihood bifurcation method is able to produce a good community structure in each network. The modularity score of our method is the highest for five networks: Dolphins, Football, C. Elegans, Erdos02 and EAT-RS and is very close to the highest modularity score (produced by Multilevel) for the other three networks. We also observe that despite producing a good modularity score, walk-likelihood bifurcation provides less number of communities compared to the other algorithms. This indicates that the communities formed using our algorithm are well separated.

Chapter 9

Conclusion and Future work

In conclusion we have developed an efficient and accurate algorithm for community detection using random-walks. The algorithm is useful in clustering both artificial and real-world networks of all range of sizes. Experiments on LFR benchmark networks show that our algorithm gives a better NMI score than all the other competing algorithms. Our algorithm also performs well on real world networks with a high value for modularity score. Although the in silico time of our algorithm is not more than 80 seconds for any of the tested networks, the time complexity of our algorithm needs to be investigated.

The future work includes three main ideas.

9.1 Community Detection for overlapping communities

Many real-world networks contain communities that can overlap. The idea is extremely natural in case of social networks where a person can belong to more than one community. They can belong to the community of their family as well as to the community of their colleagues from work. Another example is of a genes-coexpression network where a single gene can express together with more than one sets of genes.

In order to include overlapping communities, we need to modify the walk-likelihood function to also include marginal probabilities for intersections of communities ($P(n \in C_1 \cap C_2)$) in addition to marginal probabilities for single communities ($P(n \in C_1)$ and $P(n \in C_2)$).

9.2 Community Detection in a supervised learning approach

There are networks where labels of a few nodes are already known. For finding the labels of the unknown nodes, we can modify our algorithm in a simple way. The number of communities is preassigned as the number of known labels. The nodes with known labels serve as the initial belief of the algorithm. After the first step of random-walks we can fit the gauge parameter g_c mentioned in chapter 7, such that the error in labeling the known nodes is minimized. After fitting the gauge parameters, the community identity of each unknown node can be found.

9.3 Cheap method for finding number of communities in a network using Localized Random-Walks

This idea is derived from the original algorithm for community detection mentioned in appendix. We define a process called localized random-walks where the ℓ^{th} step is as follows

1. The random walker randomly teleports to a node n_i with probability $P(n_i) = K(n_i, \ell - 1)/\ell$ where $K(n_i, \ell)$ is the number of times node n_i has been visited after ℓ steps. Since, this process starts from node n_s, $K(n_j, 0) = \delta_{sj}$.

2. One step random-walks from node n_i. If node n_j is visited in this step, now $K(n_j, \ell) = K(n_j, \ell - 1) + 1$.

We observe that the localized random-walks diffuses on a community faster than the rest of the network. The diffusion on a community is of the same order as that of a normal random walk. The diffusion on the rest of the network takes $\log(\ell)$ steps when compared to ℓ steps of a normal random-walk. Thus, for a small number of steps the localized random-walks visits nodes of just one community. This property can be used in coarse-graining a large network and clustering this meta-network instead of the original large network.

Appendix A

A.1 Random Walk on a community

Consider a discrete random walk on a network with weighted edges: $\{w_{n_i n_j}\}$, where $\{w_{n_i n_j}\}$ is the rate of transmission from node n_i to n_j. At each step the random-walker jump to its nearest neighbor with probability $P(n_i \to n_j) = w_{n_j}/w_{n_i}$ where $w_{n_i} = \sum_{k \in nn(n_i)} w_k$. Assume that the network has a community C of N_c nodes and an average outward rate of $\langle w \rangle_c$, and a a set nodes $S = \{n_1, n_2, ..., n_{N_p}\}$ with average connectivity $\langle w \rangle_p$, such that $S \subset C$. The return time to the set S given there is no transition outside the community is approximately $N_c \langle w \rangle_c / N_p \langle w \rangle_p$. Assuming the exponential ansatz for the return-time distribution, the probability to return to a set of nodes $S = \{n_1, n_2, ..., n_{N_p}\}$ after exactly ℓ steps, given there is no transition outside the community is

$$P(\ell) = \frac{\mathcal{W}(S)}{\mathcal{W}(C)} e^{-\frac{\mathcal{W}(S)}{\mathcal{W}(C)}\ell} \tag{A.1}$$

where $\mathcal{W}(S)$ is $\sum_{n \in S} w_n$. Thus, $\mathcal{W}(S) = N_p \langle w \rangle_p$ and $\mathcal{W}(C) = N_c \langle w \rangle_c$. We shall refer to the sum of connectivity of a set of nodes as the weighted size of the nodes. We refer to the marked nodes $n_i \in Ps$ as pseudo-targets. It then follows that the probability to not make a return for ℓ steps (i.e. the survival probability) given no transition outside the community

$$S(\ell) = e^{-\frac{\mathcal{W}(S)}{\mathcal{W}(C)}\ell} \tag{A.2}$$

We label the above-mentioned random-walks as a multiset $(RW, \mathcal{K})$ where RW is the set of all nodes that were visited in the random-walks and $\mathcal{K}_{RW}(n)$ is the number of times the node n was visited in the random-walks. Let be $\mathcal{L}$be a function of the multiset RW and a set of nodes T such that

$$\mathcal{L}(RW,T) = \sum_{n\in(T\cap RW)} \mathcal{K}_{RW}(n) \tag{A.3}$$

Thus, $\mathcal{L}(RW,RW) = \sum_{n\in RW} \mathcal{K}_{RW}(n)$ is the total number of steps in the random-walks. Thus the random-walker makes $\mathcal{L}(RW,S)$ returns to the pseudo-target set S. If the number of steps in community C ($S \subset C$) is known, the likelihood of the event is given by

$$P(\mathcal{L}(RW,S)|\mathcal{W}(C),\mathcal{L}(RW,C)) = \frac{1}{\mathcal{L}(RW,S)!}\left(\frac{\mathcal{W}(S)}{\mathcal{W}(C)}\mathcal{L}(RW,C)\right)^{\mathcal{L}(RW,S)} e^{-(\mathcal{W}(S)/\mathcal{W}(C))\mathcal{L}(RW,C)} \tag{A.4}$$

where $\mathcal{L}(RW,C)$ is the number of steps in community C ($S \subset C$). Thus, given $\mathcal{L}(RW,S)$, $\mathcal{L}(RW,C)$ and $\mathcal{W}(S)$, and assuming uniform prior the likelihood can be maximized to obtain estimate of $\mathcal{W}(C)$

$$\widehat{\mathcal{W}}(C) = \frac{\mathcal{W}(S)\mathcal{L}(RW,C)}{\mathcal{L}(RW,S)} \tag{A.5}$$

But if the identity of all nodes which belong of the community C is not known, the quantity $\mathcal{L}(RW,C)$ is unknown. We define $\mathcal{X}$, a function of the multiset RW and community C such that $\mathcal{X}(RW,C) = \frac{\mathcal{L}(RW,C)}{\mathcal{W}(C)}$. Thus, given the random-walk multiset RW and a pseudo-target set Ssuch that $S \subset C_S$ for community C_S, we get

$$\hat{\mathcal{X}}(RW,C_S) = \frac{\mathcal{L}(RW,S)}{\mathcal{W}(S)} \tag{A.6}$$

Consider a network with m communities. Consider $\mathcal{M}$: a set of m pseudo-target sets. Each pseudo-target set $S \in \mathcal{M}$ is on a separate community C_S. The communities C_S for $S \in \mathcal{M}$ cover the whole network. Consider m separate random-walks on the network. The starting node of each random-walks is a node randomly selected from a different pseudo-target. We refer to each of these random-walks as RW_S. For each random-walks RW_S, we have $\hat{\mathcal{X}}(RW_S,C_{S'}) = \frac{\mathcal{L}(RW,S)}{\mathcal{W}(S)}$ for each set $S' \in \mathcal{M}$. We also have the constraint

$$\sum_{S'} \mathcal{L}(RW_S, C_{S'}) = \mathcal{L}(RW_S) \tag{A.7}$$

where $\mathcal{L}(RW_S)$ is the total steps of random-walks from pseudo-target set $S$is known. Thus, we have a set of m equations with m variables where m is the total number of pseudo-target sets

$$\sum_{S'} \hat{\mathcal{X}}(RW_S, C_{S'})\mathcal{W}(C_{S'}) = \mathcal{L}(RW_S) \tag{A.8}$$

Solving these we get

$$\mathcal{W}(C_S) = \sum_{S'} \mathcal{L}(RW_{S'})X^{-1}_{SS'} \tag{A.9}$$

where $X^{-1}$is the inverse of the matrix X, such that $X_{SS'} = \hat{\mathcal{X}}(RW_S, C_{S'})$.

A.2 Algorithm

The algorithm can be described in short as follows

1. Set up multiple sets of pseudo-targets such that for each set of pseudo-targets, a large portion of the pseudo-targets lie within the same community.

2. Perform random-walks process through each pseudo-target set. This process is essential in obtaining community statistics for each pseudo-target set.

3. Perform a p-value test to check whether each community of the network is covered with pseudo-targets. If yes proceed to step 4, otherwise go back to step 1. Once, the p-value test is passed, there is no need to check it again in subsequent iterations. Note that the p-value test requires results from random-walks, which is why this step is done after the random-walks process.

4. For each pair of pseudo-target sets, perform a p-value test, based on statistics of the two sets to check if they belong to the same community. If they belong to the same community combine the statistics of the two sets and merge the two as a single set.

5. Assign community identity to each node visited through random-walks process. For each community, nodes with highest confidence levels are reassigned as pseudo-targets. Go back to step 2.

The algorithm is terminated after desired accuracy is achieved. After each stage, the statistics of each pseudo-target sets are updated. Because of the structure of the algorithm, the number of communities are overestimated at the start of the algorithm. As the algorithm proceeds, the number of communities remain constant.

A.2.1 Setting up initial batch of pseudo-targets

Since the statistics of each community are based on the assumption that the cluster identity of each pseudo-target is correct, we require the initial set of pseudo-targets such that for each set of pseudo-targets, a large portion of the pseudo-targets lie within the same community. We generate a set of pseudo-targets from a random starting node n_s using the following method which we call localized random-walks (LRW). The ℓ^{th} step of localized random walk is as follows

1. The random walker randomly teleports to a node n_i randomly with probability $P(n_i) = K(n_i, \ell - 1)/\ell$ where $K(n_i, \ell)$ is the number of times node n_i has been visited after ℓ steps. Since, this process starts from node n_s, $K(n_j, 0) = \delta_{sj}$.

2. One step random-walks from node n_i. If node n_j is visited in this step, now $K(n_j, \ell) = K(n_j, \ell - 1) + 1$.

Note that teleportation to a node is not counted as a visit to the node. The LRW is halted after $\ell_{\max}$ steps where $\ell_{\max}$ depends on our the definition of community. Each node visited through LRW is marked as a pseudo-target.We do this process many times to get multiple sets of pseudo-targets. Each time the starting node n_s is randomly chosen from all nodes that have not been visited through any of the previous LRW. We refer to the set of all such pseudo-target sets as $\mathcal{M}$, The community represented by each pseudo-target set $S$for $S \in \mathcal{M}$ is referred to as C_S.

A.2.2 Random-Walks Process

The Random-walks process is performed from each pseudo-targets set sequentially. The random-walks process within a pseudo-target set S is as follows.

1. Randomly choose a pseudo-target n from the set S with probability proportional to its outward rate ($P(n) = w_n/\mathcal{W}(S)$) and start random-walks from the node n.

2. If the random-walker returns to a pseudo-target of the set S, halt the random-walks and go back to step 1

3. If $\Delta\ell_{\max}$ steps are recorded without a single return to the set S, halt the random-walks and go back to step 1

The restriction of random-walks to $\Delta\ell_{\max}$ steps ensures that the random-walks is largely localized to the community of the pseudo-target set. The random-walks process is halted if the total number of visits to any pseudo-targets is greater than $\mathcal{K}_{\max}$. Since each pseudo-target set changes with iteration, we refer to the pseudo-target set S in i^{th} iteration as $S^{(i)}$. Although, the random-walk process on a pseudo-target set consists of several random-walks terminated at either step 2 or step 3, we record all these random-walks together in multiset $RW_S^{(i)}$ where $i$stands for iteration and $S$is the starting pseudo-target set. All the random-walks from pseudo-target set Sover all iterations are stored in RW_S ($RW_S = \sum_i RW_S^{(i)}$). Let S_r be the set of all nodes visited by random-walks process from the set $S(RW_S)$ when the random-walks were terminated by returning to pseudo-targets (step 2 of the Random-Walk Process). This quantity is of particular importance because the random-walks which return to the pseudo-target set S are less likely to visit nodes outside the starting community compared to the random-walks which are terminated after $\Delta\ell_{\max}$ steps (step 3 of the Random-Walk Process).

A.2.3 Checking if pseudo-targets cover every community of a network

We assume that if each community of a network is covered with pseudo-targets, every node of the network should be accessible through short the random-walks of $\Delta\ell_{\max}$

steps. If a community does not contain even a single pseudo-target set, then the probability to reach the nodes of that community through random-walks of $\Delta\ell_{\max}$ is very low. Thus, we get the probability distribution of size of the network covered by random-walks

$$P(N|\mathcal{K}_p) = \frac{N^{-\mathcal{K}_p} \exp\left(-\frac{N_p\langle w\rangle_p}{N\langle w\rangle}\ell\right)}{\sum_{\tilde{N}=N_p}^{\infty} \tilde{N}^{-\mathcal{K}_p} \exp\left(-\frac{N_p\langle w\rangle_p}{\tilde{N}\langle w\rangle}\ell\right)} \tag{A.10}$$

where N is size of the network that is covered through random-walks from all the pseudo-targets, $\langle w\rangle$ is its average connectivity, N_p is total number of pseudo-targets across all sets,$\langle w\rangle_p$ their average connectivity and $\mathcal{K}_p$ is the total visits to all these pseudo-targets. If the size of this random-walk visited network is equal to the size of the whole network, then it means that each community of the network has at least one pseudo-target set. Since, we know the total size of the network (N_{tot}), we device a p-value test for the hypothesis that each community of the network is covered with pseudo-targets.

$$P(N \geq N_{\text{tot}}) = \sum_{N=N_{\text{tot}}}^{\infty} P(N|\mathcal{K}_p) \tag{A.11}$$

We reject the hypothesis if $P(N \geq N_{\text{tot}}) < p$.

A.2.4 Merging sets of pseudo-targets belonging to the same community

Consider two pseudo-target sets S and S'. The pseudo-target set S' is visited by all random-walks starting from S, $\mathcal{L}(RW_S, S')$ times. If pseudo-target set $S' \in C_S$ then the probability of visiting them a total of $\mathcal{K}$ times in all random-walks from the set S is given by

$$P(\mathcal{K}|\mathcal{L}(RW_S, C_S), \mathcal{W}(S'), \mathcal{W}(C_S)) = \frac{1}{\mathcal{K}!}\left(\mathcal{W}(S')\frac{\mathcal{L}(RW_S, C_S)}{\mathcal{W}(C_S)}\right)^{\mathcal{K}} e^{-\mathcal{W}(S')\mathcal{L}(RW_S,C_S)/\mathcal{W}(C_S)} \tag{A.12}$$

where $\mathcal{L}(RW_S, C_S)$ is the number of steps taken of random-walks from the set S in the community C_S. Recall that the quantity $\mathcal{X}(RW_S, C_S) = \mathcal{L}(RW_S, C_S)/\mathcal{W}(C_S)$, can be estimated if the multiset RW_S and the pseudo-target set S are known (mentioned in detail in Section 4.7). Thus we can rewrite equation 12

$$P(\mathcal{K}|\hat{\mathcal{X}}(RW_S, C_S), \mathcal{W}(S')) = \frac{1}{\mathcal{K}!}\left(\mathcal{W}(S')\hat{\mathcal{X}}(RW_S, C_S)\right)^{\mathcal{K}} e^{-\mathcal{W}(S')\hat{\mathcal{X}}(RW_S, C_S)} \tag{A.13}$$

We device a p-value test for the hypothesis that the pseudo-target set S' belong to the community C_S. $(H(S' \in C_S))$.

$$P(\mathcal{K} \leq \mathcal{L}(RW_S, S')) = \sum_{\mathcal{K}=0}^{\mathcal{L}(RW_S, S'))} P(\mathcal{K}'|\hat{\mathcal{X}}(RW_S, C_S), \mathcal{W}(S')) \tag{A.14}$$

We reject the hypothesis $H(S' \in C_S)$ if $P(\mathcal{K}' \leq \mathcal{K}(S, S')) < p$. For each pair S and S' we test both $H(S' \in C_S)$ and $H(S \in C_{S'})$. If both these hypothesis are true it means that C_S and $C_{S'}$ indicate the same community $(C_i = C_j)$. If $H(S' \in C_S)$ is true but not $H(S \in C_{S'})$ then it indicates that $C_{S'} \subset C_S$. If both the hypothesis are rejected then C_S and $C_{S'}$ are separate communities. If either condition hold true we merge the pseudo-target sets S and S' as a single pseudo-target set and combine their statistics of all iterations. ($\tilde{S}^{(i)} = S^{(i)} \cup S'^{(i)}$ for $\tilde{S} = S \cup S'$). This test is done in loop over all pairs of sets till the hypothesis is rejected by each pair.

A.2.5 Assigning each visited node to a community

Since the random-walks from each pseudo-target set S are not perfectly restricted to the community C_S, it is very likely that a node is visited by random-walks from multiple sets. If a node is visited in random-walks through only 1 pseudo-target S set then it is assigned the community C_S. Consider all pseudo-target sets S' such that the node n is visited at least once in random-walks from set S' $(\mathcal{K}_{RW_{S'}}(n) > 0)$. Let the set of all such pseudo-target sets be T. The possible community assignments the node n are all communities $C_{S'}$ such that $S' \in T$. Now, if for at least one pseudo-target set $S \in T$, the node n is visited through random-walks from S which return to pseudo-targets $(n \in S_r)$, the probability that the node $n \notin C_S$ is very low. Hence if such a set S exists, we remove all sets S' from T if $n \notin S_r$. Now, if the node $n \in C_{\tilde{S}}$ then the node is visited from the random-walks from all other sets S only after transitioning into community $C_{\tilde{S}}$. Thus, if the node $n \in C_{\tilde{S}}$ the probability of it is visited $k(n, S)$ in random-walks from each set S is given by

$$P(\{\mathcal{K}_{RW_S}(n)\}|n \in C_{\tilde{S}}) = \prod_{S}^{S\in T} \frac{1}{\mathcal{K}_{RW_S}(n)!} \left(w_n \hat{\mathcal{X}}(RW_S, C_{\tilde{S}})\right)^{-\mathcal{K}_{RW_S}(n)} e^{-w_n \hat{\mathcal{X}}(RW_S, C_{\tilde{S}})} \tag{A.15}$$

where $q(S, \tilde{S}) = \ell_{S\tilde{S}}/\mathcal{W}(C_{\tilde{S}})$ where $\ell_{S\tilde{S}}$ is the number of steps of random-walks starting from the set S in community $\ell_{S\tilde{S}}$. But since $\hat{\ell}_{S\tilde{S}} = \frac{\mathcal{W}(C_{\tilde{S}})}{\mathcal{W}(S)}\mathcal{K}(S, \tilde{S})$ as mentioned in equation 8, $q(S, \tilde{S}) = \mathcal{K}(S, \tilde{S})/\mathcal{W}(S)$. Note that this is not applicable for $S = \tilde{S}$ the choice of the nodes which the pseudo-target set $\tilde{S}$ is not independent of previous the random-walks from $\tilde{S}$. Hence, for $S = \tilde{S}$ we use $q(\tilde{S}, \tilde{S}) = \sum_i \frac{\mathcal{K}^{(i)}(\tilde{S},\tilde{S})}{\mathcal{W}(S)(\tilde{S}^{(i)})}$ as mentioned before. Thus, assuming a uniform prior to each community, by Bayes' theorem we have

$$P(n \in C_{\tilde{S}}|\{k(n, S)\}) = \frac{P(\{k(n, S)\})|n \in C_{\tilde{S}}}{\sum_{S'}^{S'\in T} P(\{k(n, S)\})|n \in C_{S'}} \tag{A.16}$$

The node n is assigned to the community $C_{\tilde{S}}$ for the probability $P(n \in C_{\tilde{S}}|\{k(n, S)\})$ is the highest. Since the accuracy of pseudo-targets is essential in estimating community statistics, only the nodes with high confidence levels are reassigned as pseudo-targets. Thus, each node n is reassigned as a pseudo-target of the set $\tilde{S}$ if $n \in \tilde{S}_r$, $P(n \in C_{\tilde{S}}|\{k(n, S)\}) > P(n \in C_{S'}|\{k(n, S)\})$ for all $S' \neq \tilde{S}$ and $P(n \in C_{\tilde{S}}|\{k(n, S)\}) > p$ for a threshold p.

The pseudo-target sets $S^{(i)}$ in previous iterations are also important in finding certain statistics. Therefore, we purify them retroactively.

$$S^{(i)} = S_o^{(i)} \cap S \tag{A.17}$$

where $S_o^{(i)}$ is the first version of $S^{(i)}$.

A.2.6 Halting Criteria

We halt the algorithm if there is no merging of communities for $t_{\max}$ straight iterations. This ensures that each community obtained is well separated.

A.2.7 Statistics

Estimating $\mathcal{X}(RW_S, C_{S'})$: Recall that $\hat{\mathcal{X}}(RW_S, C_{S'}) = \frac{\mathcal{L}(RW_S,S')}{\mathcal{W}(S')}$ as mentioned before. This estimate is unbiased if the random-walks RW_S and the sampling of pseudo-targets for the set S' are independent. But since the choice of pseudo-targets in a community C_S is based on the nodes that were visited in the random-walks RW_S, this is not applicable for $S = S'$. The random-walks in i^{th} iteration are independent of the random-walks in previous iterations. Thus, $\hat{\mathcal{X}}(RW_S^{(i)}, C_S) = \frac{\mathcal{L}(RW_S^{(i)}, C_S)}{\mathcal{W}(S^{(i)})}$. Also, the nodes visited in the random-walks $RW_S^{(i)}$ which belong to community C_S can serve as pseudo-targets for the random-walks in the previous iterations. Hence we can estimate for the quantity $\mathcal{X}(\tilde{RW}_S^{(i)}, C_S)$ where $\tilde{RW}_S^{(i)} = \sum_{j=1}^{i} RW_S^{(j)}$ refers to all the random-walks starting from pseudo-target S till iteration i.

$$\hat{\mathcal{X}}(\tilde{RW}_S^{(i-1)}, C_S) = \frac{\mathcal{L}(\tilde{RW}_S^{(i-1)}, \tilde{S^{(i)}})}{\mathcal{W}(\tilde{S^{(i)}})} \tag{A.18}$$

where $\tilde{S^{(i)}} = S \cap RW_S^{(i)}$ is the set of all nodes visited in $RW_S^{(i)}$ which are assigned as the nodes of the community C_S with high confidence. Since $\mathcal{L}(\tilde{RW}_S^{(i)}, C_S) = \mathcal{L}(\tilde{RW}_S^{(i-1)}, C_S) + \mathcal{L}(RW_S^{(i)}, C_S)$, we assume $\hat{\mathcal{X}}(\tilde{RW}_S^{(i)}, C_S) = \hat{\mathcal{X}}(\tilde{RW}_S^{(i-1)}, C_S) + \hat{\mathcal{X}}(RW_S^{(i)}, C_S)$. After Random-Walks process, the community identity of all nodes visited is the latest random-walks is unknown till step 4 of the algorithm which is essential in finding the set $\tilde{S^{(i)}}$. Hence after the random-walk process we use

$$\hat{\mathcal{X}}(\tilde{RW}_S^{(t)}, C_S) = \hat{\mathcal{X}}(\tilde{RW}_S^{(t-2)}, C_S) + \hat{\mathcal{X}}(RW_S^{(t-1)}, C_S) + \hat{\mathcal{X}}(RW_S^{(t)}, C_S) \tag{A.19}$$

where t is the latest iteration. After each node visited in the random-walks $RW_S^{(t)}$ is assigned to a community we use

$$\hat{\mathcal{X}}(\tilde{RW}_S^{(t)}, C_S) = \hat{\mathcal{X}}(\tilde{RW}_S^{(t-1)}, C_S) + \hat{\mathcal{X}}(RW_S^{(t)}, C_S) \tag{A.20}$$

Thus, we have the estimate for $\hat{\mathcal{X}}(RW_S, C_{S'})$ for all sets S and S'. Using equation 9, we can estimate the weighted size of each community C_S

$$\mathcal{W}(C_S) = \sum_{S'} \mathcal{L}(RW_{S'}) X_{SS'}^{-1} \tag{A.21}$$

where $X_{SS'}{=}\hat{\mathcal{X}}(RW_S, C_{S'})$. To find the number of nodes in each community we estimate $\langle w \rangle$: the average connectivity of each community as $\mathcal{W}(C) = N_C \langle w \rangle_C$. Consider a set $T_{w_i} \subset C$ which contains all nodes with outward rate w_i in community C. The probability that the random walker visits the set $\mathcal{K}_{w_i}$ times in ℓsteps in the community C is given by

$$P(\mathcal{K}_{w_i}, \ell | p_{w_i}) = \frac{1}{\mathcal{K}_{w_i}!} \left(\ell p_{w_i} \frac{w_i}{\langle w \rangle} \right)^{\mathcal{K}_{w_i}} e^{-\ell p_{w_i} w_i / \langle w \rangle} \tag{A.22}$$

where $p_{w_i} = N_{w_i}/N_c$ and N_{w_i} is the number of nodes in the set T_{w_i}. Note that $\ell = \sum_i \mathcal{K}_{w_i}$. The maximum likelihood estimate of p_{w_i} is $\frac{\mathcal{K}_{w_i}/w_i}{\ell/\langle w \rangle}$. But $\sum_i p_{w_i} = 1$, where the summation is over all such sets T_{w_i}. This gives us

$$\langle w \rangle = \frac{\sum_i \mathcal{K}_{w_i}}{\sum_i \mathcal{K}_{w_i}/w_i} \tag{A.23}$$

For a community C, we calculate $\mathcal{K}_{w_i}$ from all set of nodes of the network which have outward rate w_i and have been assigned the community C.

A.3 Biased Random Walks

The ℓ^{th} step of biased random walk is as follows

1. The random walker randomly teleports to a node n_i randomly with probability $P(n_i) = K_i(\ell-1)/\ell$ where $K_i(\ell)$ is the number of times node n_i has been visited in after ℓ step.

2. One step random walk from node n_i. If node n_j is visited in this step, now $K_j(\ell) = K_j(\ell-1) + 1$.

Note that teleportation to a node is not counted as a visit to the node. The initial state of this process is kept to be a single node n_s: $K_j(0) = \delta_{sj}$. Thus, the probability of visiting node n_i on ℓ^{th} step is given by

$$P(n_i, \ell | K) = \sum_{j \in nn(i)} \frac{K_j(\ell-1)}{\ell} \frac{w_{ji}}{w_j} \tag{A.24}$$

where $nn(i)$ refers to the nearest neighbors of node n_i. Thus,

$$\bar{P}(n_i, \ell | K(\ell-1)) = \sum_{j \in nn(i)} \frac{\bar{K}_j(\ell-1)}{\ell} \frac{w_{ji}}{w_j} \tag{A.25}$$

where $\bar{P}(n_i, \ell|K(\ell-1))$ and $\bar{K}_j(\ell-1)$ are the expectation values. But

$$\bar{P}(n_i, \ell|K(\ell-1)) = \sum_{\{K'(\ell-1)\}} P(n_i, \ell|K'(\ell-1))P(K'(\ell-1)|n_s) = P(n_i, \ell|n_s) \quad \text{(A.26)}$$

Here, $\{K'(\ell-1)\}$ refers to all possible configurations of the list $K(\ell-1)$. $P(K'(\ell-1)|n_s)$ is the probability to get the list $\{K_j(\ell-1)\} = \{K'_j(\ell-1)\}$ after $\ell-1$ steps if the biased random walk started from node n_s at $\ell = 0$ ($K_j(0) = \delta_{sj}$) and $P(n_i, \ell|n_s)$ is the probability to visit node n_i at step ℓ with the same initial condition. Since $\bar{K}_j(\ell)$ is the expectation of the total number of times node n_j has been visited till ℓ steps, it is just the sum of probability of visiting the node in each step till step ℓ ($\bar{K}_j(\ell) = \sum_{l'=0}^{\ell} P(n_j, l'|n_s)$). This gives us a forward equation for P

$$P(n_i, \ell+1|n_s) = \frac{\ell P(n_i, \ell|n_s)}{\ell+1} + \sum_{j\in nn(i)} \frac{P(n_j, \ell|n_s)}{\ell+1}\frac{w_{ji}}{w_j} \quad \text{(A.27)}$$

Thus, the steady state is the same as that of normal random walks $P(n_i|n_s) = \frac{w_i}{\sum_j' w_j'}$. Consider a network with comprising of a multiple communities with the random walker initially in community C ($n_s \in C$). Let $P_{out}(\ell) = \sum_i^{n_i \notin C} P(n_i, \ell|n_s)$ and $P_{in}(\ell) = \sum_i^{n_i \in C} P(n_i, \ell|n_s)$. From equation 10 we get

$$\begin{aligned}
P_{out}(\ell+1) &= \frac{\ell P_{out}(\ell)}{\ell+1} + \sum_i^{n_i\notin C} \sum_{j\in nn(i)} \frac{P(n_j, \ell|n_s)}{\ell+1}\frac{w_{ji}}{w_j} \\
&= \frac{\ell P_{out}(\ell)}{\ell+1} + \sum_j \sum_i^{n_i\notin C} \frac{P(n_j, \ell|n_s)}{\ell+1}\frac{w_{ji}}{w_j} \\
&= \frac{\ell P_{out}(\ell)}{\ell+1} + \sum_j^{n_j\in C} \sum_i^{n_i\notin C} \frac{P(n_j, \ell|n_s)}{\ell+1}\frac{w_{ji}}{w_j} + \sum_j^{n_j\notin C} \sum_i^{n_i\notin C} \frac{P(n_j, \ell|n_s)}{\ell+1}\frac{w_{ji}}{w_j}
\end{aligned} \quad \text{(A.28)}$$

Note that if $j \notin nn(i)$ $w_{ij} = 0$. Assuming that $P(n_i, \ell|n_s) \approx \frac{w_i P_{in}(\ell)}{\sum_j^{n_j\in C} w_j}$ for $n_i \in C$ and $P(n_i, \ell|n_s) \approx \frac{w_i P_{out}(\ell)}{\sum_j^{n_j\notin C} w_j}$ for $n_i \notin C$, we get

$$P_{out}(\ell+1) = \frac{1}{\ell+1}\left(\ell P_{out}(\ell) + \sum_j^{n_j\in C} \sum_i^{n_i\notin C} \frac{P_{in}(\ell) w_{ji}}{\sum_k^{n_k\in C} w_k} + \sum_j^{n_j\notin C} \sum_i^{n_i\notin C} \frac{P_{out}(\ell) w_{ji}}{\sum_k^{n_k\notin C} w_k}\right) \quad \text{(A.29)}$$

Let $Q = \sum_j^{n_j \in C} \sum_i^{n_i \notin C} \frac{w_{ji}}{\sum_k^{n_k \in C} w_k}$ and $Q_B = \sum_j^{n_j \notin C} \sum_i^{n_i \in C} \frac{w_{ji}}{\sum_k^{n_k \notin C} w_k}$. Thus, we get

$$\begin{aligned} P_{out}(\ell+1) &= \frac{1}{\ell+1}\left(\ell P_{out}(\ell) + QP_{in}(\ell) + (1-Q_B)P_{out}(\ell)\right) \\ &= \frac{1}{\ell+1}\left(\ell P_{out}(\ell) + Q(1-P_{out}(\ell)) + (1-Q_B)P_{out}(\ell)\right) \\ P_{out}(\ell+1) - P_{out}(\ell) &= -\frac{1}{\ell+1}(Q+Q_B)P_{out}(\ell) + \frac{Q}{\ell+1} \end{aligned} \tag{A.30}$$

Solving this in continuous time we get

$$P_{out}^{BRW}(\ell) = \frac{Q}{Q+Q_B}\left(1 - e^{-(Q+Q_B)\log(\ell)}\right) \tag{A.31}$$

For a normal random walk the forward equation is

$$P^{NRW}(n_i, \ell+1|n_s) - P^{NRW}(n_i, \ell|n_s) = \sum_{j \in nn(i)} \frac{w_{ji}}{w_j} P^{NRW}(n_j, \ell|n_s) - P^{NRW}(n_i, \ell|n_s) \tag{A.32}$$

where $P^{NRW}(n_i, \ell|n_s)$ is the probability of visiting node n_i at step ℓ in normal random walk, if the random walker started from n_s at $\ell = 0$. Applying equation 16 to the network with multiple communities as discussed above, we get

$$P_{out}^{NRW}(\ell+1) - P_{out}^{NRW}(\ell) = -(Q+Q_B)P_{out}^{NRW}(\ell) + Q \tag{A.33}$$

where $P_{out}^{NRW}(\ell) = \sum_i^{n_i \notin C} P^{NRW}(n_i, \ell|n_s)$ and $P_{in}^{NRW}(\ell) = \sum_i^{n_i \in C} P^{NRW}(n_i, \ell|n_s)$. In continuous time limit we get

$$P_{out}^{NRW}(\ell) = \frac{Q}{Q+Q_B}\left(1 - e^{-(Q+Q_B)\ell}\right) \tag{A.34}$$

This equation is same as equation 16 with ℓ instead of $\log(\ell)$. Thus, biased random walk slows time of diffusing outside the starting community to logarithmic scale.

Alternate Method for logarithmic behavior: The forward equation for biased random walks is

$$P^B(n_i, \ell+1|n_s) - P^B(n_i, \ell|n_s) = \frac{1}{\ell+1}\left(\sum_{j \in nn(i)} \frac{w_{ji}}{w_j} P^B(n_j, \ell|n_s) - P^B(n_i, \ell|n_s)\right) \tag{A.35}$$

In case of normal random walk the forward equation is

$$P^N(n_i, \ell+1|n_s) - P^N(n_i, \ell|n_s) = \sum_{j \in nn(i)} \frac{w_{ji}}{w_j} P^N(n_j, \ell|n_s) - P^N(n_i, \ell|n_s) \tag{A.36}$$

which can be written in continuous time limit as

$$\frac{d}{d\ell}P^N(n_i,\ell|n_s) = \sum_{j\in nn(i)} \frac{w_{ji}}{w_j}P^N(n_j,\ell|n_s) - P^N(n_i,\ell|n_s) \tag{A.37}$$

Assume that $P^B(n_i,\ell|n_s) = P^N(n_i,\log(\ell+1)|n_s)+\epsilon(n_i,\ell|n_s)$. Note that $\epsilon(n_i,0|n_s) = 0$ as P^B and P^N have the same initial conditions. Thus,

$$\begin{aligned} P^B(n_i,\ell+1|n_s) - P^B(n_i,\ell|n_s) &= \epsilon(n_i,\ell+1|n_s) - \epsilon(n_i,\ell|n_s) \\ &+ \log\left(\frac{\ell+2}{\ell+1}\right)\left(\frac{P^N(n_i,\log(\ell+2)|n_s) - P^N(n_i,\log(\ell+1)|n_s)}{\log\left(\frac{\ell+2}{\ell+1}\right)}\right) \\ &\approx \epsilon(n_i,\ell+1|n_s) - \epsilon(n_i,\ell|n_s) + \log\left(\frac{\ell+2}{\ell+1}\right)\frac{d}{dl'}P^N(n_i,l'|n_s)\bigg|_{l'=\log(\ell+1)} \end{aligned} \tag{A.38}$$

Substituting this in equation 19 and substituting $\frac{d}{dl'}P^N(n_i,l'|n_s)$ with RHS of equation 21, we get

$$\epsilon(n_i,\ell+1|n_s) - \epsilon(n_i,\ell|n_s) = g(\ell)H(P^N)(n_i,\log(\ell+1)|n_s) + H(\epsilon)(n_i,\ell|n_s) \tag{A.39}$$

where $H(f)(n_i,\ell|n_s) = \sum_{j\in nn(i)} \frac{w_{ji}}{w_j} f(n_j,\ell|n_s) - f(n_i,\ell|n_s)$ and $g(\ell) = \left(\frac{1}{\ell+1} - \log\left(1+\frac{1}{\ell+1}\right)\right)$. If n_s and n_i are in different communities the flux towards n_i at lower ℓ is small which makes $H(P^N)(n_i,\log(\ell+1)|n_s)$ negligible at small ℓ. At larger ℓ, $g(\ell)$ becomes small. Thus the term $\epsilon(n_i,\ell|n_s) \approx 0$ and $P^B(n_i,\ell|n_s) \approx P^N(n_i,\log(\ell+1)|n_s)$. Thus, $P^B_{out}(\ell) = P^N_{out}(\log(\ell+1))$ where $P_{out}(\ell) = \sum_j^{n_j\notin C} P(n_j,\ell|n_s)$ and C is the community of n_s.

If n_i and n_s are in the same community $H(P^N)(n_i,\log(\ell+1)|n_s)$ is considerable. When, $g(\ell)$ becomes negligible the diffusion within the starting community has already taken place. Thus $P^B(n_i,\ell|n_s) > P^N(n_i,\log(\ell+1)|n_s)$ if n_i and n_s are in the same community.

References

[1] Aditya Ballal, Caroline Laurendon, Melissa Salmon, Maria Vardakou, Jitender Cheema, Marianne Defernez, Paul E O'Maille, and Alexandre V Morozov. Sparse Epistatic Patterns in the Evolution of Terpene Synthases. *Molecular Biology and Evolution*, 37(7):1907–1924, 04 2020.

[2] Aditya Ballal, Constantin D. Malliaris, and Alexandre V. Morozov. *Molecular Phenotypes as Key Intermediates in Mapping Genotypes to Fitness*, pages 15–40. Springer International Publishing, Cham, 2020.

[3] Tholl D. Terpene synthases and the regulation, diversity and biological roles of terpene metabolism. 2006.

[4] Tholl D. 2015. Biosynthesis and biological functions of terpenoids in plants. 2015.

[5] Dokarry M, Laurendon C, and O'Maille PE. Automating gene library synthesis by structure-based combinatorial protein engineering: examples from plant sesquiterpene synthases. 2012.

[6] Salmon M, Vardakou M Laurendon C, Cheema J, Defernez M, Green S, Faraldos JA, and O'Maille PE. Emergence of terpene cyclization in *Artemisia annua*. 2015.

[7] Larkin MA, Blackshields G, Brown NP, Chenna R, McGettigan PA, McWilliam H, Valentin F, Wallace IM, Wilm A, and Lopez R et al. Clustal w and clustal x version 2.0. 2007.

[8] Bork P. Letunic I. Interactive tree of life (itol) v4: recent updates and new developments. 2019.

[9] O'Maille PE, Chappell J, and Noel JP. A single-vial analytical and quantitative gas chromatography-mass spectrometry assay for terpene synthases. 2004.

[10] Garrett SR, Morris RJ, and O'Maille PE. Steady-state kinetic characterization of sesquiterpene synthases by gas chromatography-mass spectroscopy. 2012.

[11] Vardakou M, Salmon M, Faraldos JA, and O'Maille PE. 2014. Comparative analysis and validation of the malachite green assay for the high throughput biochemical characterization of terpene synthases. 2014.

[12] Nelson PC and Bromberg S. Radosavljevic and. Biological physics : energy, information, life. 2004.

[13] Bozlee BJ. Reformulation of the michaelis-menten equation: How enzyme-catalyzed reactions depend on gibbs energy. 2007.

[14] Binder K and Young AP. Spin-glasses - experimental facts, theoretical concepts, and open questions. 1986.

[15] Mezard M and Montanari A. Information, physics, and computation. 2009.

[16] Haq O, Levy RM, Morozov AV, and Andrec M. Pairwise and higher-order correlations among drug-resistance mutations in hiv-1 subtype b protease. 2009.

[17] Weigt M, White RA, Szurmant H, Hoch JA, and Hwa T. Identification of direct residue contacts in protein-protein interaction by message passing. 2009.

[18] Morcos F, Pagnani A, Lunt B, Bertolino A, Marks DS, Sander C, Zecchina R, Onuchic JN, Hwa T, and Weigt M. Direct-coupling analysis of residue coevolution captures native contacts across many protein families. 2011.

[19] Haq O, Andrec M, Morozov AV, and Levy RM. Correlated electrostatic mutations provide a reservoir of stability in hiv protease. 2012.

[20] Tibshirani R. The lasso method for variable selection in the cox model. 1997.

[21] Bishop CM. Pattern recognition and machine learning. 2006.

[22] Weinreich DM, Watson RA, and Chao L. Perspective: Sign epistasis and genetic constraint on evolutionary trajectories. 2005.

[23] Carneiro M and Hartl DL. Colloquium papers: Adaptive landscapes and protein evolution. 2010.

[24] Manhart M and Morozov AV. Statistical physics of evolutionary trajectories on fitness landscapes. 2014.

[25] Poelwijk FJ, Tanase-Nicola S, Kiviet DJ, and Tans SJ. Reciprocal sign epistasis is a necessary condition for multi-peaked fitness landscapes. 2011.

[26] Zhang Y. I-tasser server for protein 3d structure prediction. 2008.

[27] Yang J, Yan R, Roy A, Xu D, Poisson J, and Zhang Y. The i-tasser suite: protein structure and function prediction. 2015.

[28] Manhart M and Morozov AV. Path-based approach to random walks on networks characterizes how proteins evolve new functions. 2013.

[29] Manhart M and Morozov AV. Protein folding and binding can emerge as evolutionary spandrels through structural coupling. 2015a.

[30] Shannon P, Markiel A, Ozier O, Baliga NS, Wang JT, Ramage D, Amin N, Schwikowski B, and Ideker T. Cytoscape: a software environment for integrated models of biomolecular interaction networks. 2003.

[31] Doncheva NT, Klein K, Domingues FS, and Albrecht M. Analyzing and visualizing residue networks of protein structures. 2011.

[32] Wells JA. Additivity of mutational effects in proteins. 1990.

[33] Serrano L, Day AG, and Fersht AR. Step-wise mutation of barnase to binase. a procedure for engineering increased stability of proteins and an experimental analysis of the evolution of protein stability. 1993.

[34] Zhang XJ, Baase WA, Shoichet BK, Wilson KP, and Matthews BW. Enhancement of protein stability by the combination of point mutations in t4 lysozyme is additive. 1995.

[35] Thorn KS and Bogan AA. Asedb: a database of alanine mutations and their effects on the free energy of binding in protein interactions. 2001.

[36] Istomin AY, Gromiha MM, Vorov OK, Jacobs DJ, and Livesay DR. New insight into long-range nonadditivity within protein double-mutant cycles. 2008.

[37] Zeldovich KB, Chen P, and Shakhnovich EI. Protein stability imposes limits on organism complexity and speed of molecular evolution. 2007.

[38] Shakhnovich EI. Serohijos AW. Contribution of selection for protein folding stability in shaping the patterns of polymorphisms in coding regions. 2014.

[39] Manhart M and Morozov AV. Scaling properties of evolutionary paths in a biophysical model of protein adaptation. 2015b.

[40] Bershtein S, Serohijos AW, and Shakhnovich EI. Bridging the physical scales in evolutionary biology: from protein sequence space to fitness of organisms and populations. 2017.

[41] Phillips PC. Epistasis–the essential role of gene interactions in the structure and evolution of genetic systems. 2008.

[42] Chou HH, Chiu HC, Segre D Delaney NF, and Marx CJ. Diminishing returns epistasis among beneficial mutations decelerates adaptation. 2011.

[43] Weinreich DM, Lan Y, Wylie CS, and Heckendorn RB. Should evolutionary geneticists worry about higher-order epistasis? 2013.

[44] Adams RM, Kinney JB, Walczak AM, and Mora T. Epistasis in a fitness landscape defined by antibody-antigen binding free energy. 2019.

[45] Marks DS, Hopf TA, and Sander C. Protein structure prediction from sequence variation. 2012.

[46] Ferguson AL, Mann JK, Ndung'u T Omarjee S, Walker BD, and Chakraborty AK. Translating hiv sequences into quantitative fitness landscapes predicts viral vulnerabilities for rational immunogen design. 2013.

[47] Hart GR and Ferguson AL. Empirical fitness models for hepatitis c virus immunogen design. 2015.

[48] Neuwald AF. Sciencedirect gleaning structural and functional information from correlations in protein multiple sequence alignments. 2016.

[49] Haldane A and Morozov AV. Manhart M. Biophysical fitness landscapes for transcription factor binding sites. 2014.

[50] Newman M. E. J. Fast algorithm for detecting community structure in networks. *Phys. Rev. E*, 69:066133, Jun 2004.

[51] Usha Nandini Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, 76(3), Sep 2007.

[52] M. E. J. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical Review E*, 74(3), Sep 2006.

[53] Pascal Pons and Matthieu Latapy. Computing communities in large networks using random walks. In pInar Yolum, Tunga Güngör, Fikret Gürgen, and Can Özturan, editors, *Computer and Information Sciences - ISCIS 2005*, pages 284–293, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[54] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, Oct 2008.

[55] Willow B. Kion-Crosby and Alexandre V. Morozov. Rapid bayesian inference of global network statistics using random walks. *Phys. Rev. Lett.*, 121:038301, Jul 2018.

[56] Pentti Paatero, Unto Tapper, Pasi Aalto, and Markku Kulmala. Matrix factorization methods for analysing diffusion battery data. *Journal of Aerosol Science*, 22:S273–S276, 1991. Proceedings of the 1991 European Aerosol Conference.

[57] Pentti Paatero and Unto Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2):111–126, 1994.

[58] Chris Ding, Xiaofeng He, and Horst D. Simon. *On the Equivalence of Nonnegative Matrix Factorization and Spectral Clustering*, pages 606–610.

[59] C. Boutsidis and E. Gallopoulos. Svd based initialization: A head start for nonnegative matrix factorization. *Pattern Recognition*, 41(4):1350–1362, 2008.

[60] Zhao Yang, René Algesheimer, and Claudio J. Tessone. A comparative analysis of community detection algorithms on artificial networks. *Scientific Reports*, 6(1):30750, Aug 2016.

[61] Andrea Lancichinetti, Santo Fortunato, and Filippo Radicchi. Benchmark graphs for testing community detection algorithms. *Physical Review E*, 78(4), Oct 2008.

[62] Lusseau D. The emergent properties of a dolphin social network. 2003.

[63] D. Knuth. The stanford graphbase - a platform for combinatorial computing. 1993.

[64] M. Girvan and M. E. J. Newman. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences*, 99(12):7821–7826, 2002.

[65] PABLO M. GLEISER and LEON DANON. Community structure in jazz. *Advances in Complex Systems*, 06(04):565–573, Dec 2003.

[66] White JG, Southgate E, Thomson JN, and Brenner S. The structure of the nervous system of the nematode caenorhabditis elegans. *Philos Trans R Soc Lond B Biol Sci.*, 1986 Nov 12.

[67] Jérôme Kunegis. KONECT – The Koblenz Network Collection. In *Proc. Int. Conf. on World Wide Web Companion*, pages 1343–1350, 2013.

[68] Ryan A. Rossi and Nesreen K. Ahmed. The network data repository with interactive graph analytics and visualization. In *AAAI*, 2015.

[69] G. R. Kiss, C. Armstrong, R. Milroy, and J. Piper. An associative thesaurus of English and its computer analysis. In A. J. Aitkin, R. W. Bailey, and N. Hamilton-Smith, editors, *The computer and literary studies.* University Press, Edinburgh, UK, 1973.

[70] J. Gehrke, P. Ginsparg, and J. Kleinberg. Overview of the 2003 kdd cup. *SIGKDD Explor.*, 5:149–151, 2003.