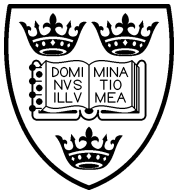
Probabilistic Bias in Genotype-Phenotype Maps



Kamaludin Dingle
St John's College
University of Oxford

A thesis submitted for the degree of
Doctor of Philosophy

Trinity 2014

For my father

Acknowledgements

First and foremost, I would like to thank Ard Louis for his supervision, and for introducing me to the fascinating questions that we have pursued during my time as a DPhil candidate. Without the DPhil companionship of fellow student Steffen Schaper, undertaking research in biological evolution within a centre for theoretical physics would have been somewhat solitary. I am grateful to Steffen for his good company, and additionally for his help in many aspects of the computational side of my work. More recently, Benjamin Frot has been a patient and harworking co-worker in our investigations into compression algorithms. Hector Zenil has also been an enthusiastic collaborator, and helped in getting to grips with the basic notions of Kolmogorov complexity. In this latter respect, I would also like to thank Peter Gács and Paul Vitányi. Iain Johnston, Sebastian Ahnert, and Sam Greenbury have formed valuable partners in discussions on complexity and evolution. Indeed, this thesis builds on some of Iain's DPhil work.

The support and guidance I have received from the Systems Biology Doctoral Training Centre (DTC) has been invaluable, and special thanks goes to Gail Preston. I am also grateful to Nick Jones for suggesting the DTC to me, without whom I probably would not have undertaken this thesis. Financial support from the EPSRC is also appreciated.

Finally, my wife Asma Rahman deserves much thanks for her continued support, encouraging degree of interest in my work, and putting up with a distracted ("PhD-ing") husband through the years of his doctorate.

Abstract

Among the most fundamental features shared by all organisms is the mapping of information encoded in *genotypes* (genetic material) to generate *phenotypes* (biological structures, functions, and traits). Hence, elucidating the structure of *genotype-phenotype (GP) maps* is important for understanding evolution and biology.

While it is known that often GP maps are highly degenerate with many different genotypes adopting the same phenotype, the distribution of genotypes over phenotypes is less well studied. In this thesis we investigate the question of the distribution of genotypes over phenotypes, or put differently the distribution of *neutral set sizes* (NSS), where a neutral set is the collection of all genotypes in a GP map which map to the same phenotype. We focus on examining *phenotypic bias* in GP maps, where some phenotypes have disproportionally large NSS as compared to others. We find phenotypic bias to be ubiquitous in the broad range of GP maps that we analyse, from the genetic code up to molecular RNA to a model of neuronal connections, and hence we hypothesise bias to be a common property of GP maps. Further, we also consider the implications that this bias has for evolutionary outcomes, and we argue that bias is a significant influencing factor in determining evolutionary outcomes.

Finally, we propose a method to predict a phenotype's NNS via estimating the phenotype's structural complexity, without using detailed knowledge about the specifics of the relevant GP map. We achieve this via a novel application of *algorithmic information theory* and especially Levin's *coding theorem*.

Contents

Chapter 1

Introduction

Despite the flourishing diversity of life, biological organisms share many basic features. Among the most fundamental of these features is the mapping of information encoded in *genotypes* (genetic material) to generate *phenotypes* (biological structures, functions, and traits). As such, in working towards a theoretical understanding of evolution and biology, it is important to elucidate the mathematical structure of these *genotype-phenotype (GP) maps*.

From an abstract perspective, a GP map can be viewed as a map from a collection of sequences (genotypes) to a collection of traits or structures (phenotypes). Framed this way, one of the most basic questions we can ask is: How are the genotypes distributed between phenotypes? More informally, if some GP map is 'fed' a random string of DNA, what probabilities should we assign to the various possible associated phenotypes? In particular, should we expect a roughly even probability for each phenotype, or might some probabilistic bias exist such that some phenotypes are far more likely than others? Perhaps surprisingly, these questions have not (to our knowledge) been clearly and systematically expounded. This thesis marks a step towards exploring these questions.

As mutations occur at the level of the genotype, while natural selection acts at the level of the phenotype, the structure of GP maps determines the interplay between these two essential components of evolution (i.e. mutation and selection). With this in mind, beyond the abstract questions posed above we also consider the implications of bias for evolutionary outcomes. Specifically, we are interested in the extent to

which evolutionary outcomes are constrained and conform to the bias (if it exists in a map). In this regard, the two main contentions which we make in the following pages are that:

(a) Probabilistic bias, or *phenotypic bias* [15], is ubiquitous in GP maps (of the form we consider), such that bias should be assumed as a 'null model' prediction in GP maps; and

(b) Bias affects evolutionary dynamics and outcomes, to the degree that phenotypes 'favoured' by the bias will be overrepresented in nature.

These two contentions are expounded in Chapter 2, which, by surveying the background, motivation, and earlier works related to phenotypic bias, forms the true Introduction to this thesis, even if not by name.

Moving on to Chapter 3, we study one GP map in detail (the RNA sequence to secondary structure map), determining the form of the distribution of genotypes over phenotypes (and hence phenotype probabilities), and additionally other key quantitative properties of the map. Perhaps the most important result of Chapter 3 is our finding that natural and random RNA secondary structures have very similar distributions in terms of the numbers of sequences per structure. This result suggests that natural phenotype abundances can be accurately predicted by knowing the probability of generating a structure from a random genotype, and highlights the importance of the structure of the GP map.

Having argued that probabilistic bias plays an important role in determining evolutionary outcomes, in Chapter 4 we consider if it is possible to predict phenotype probabilities just from the phenotype structures, without recourse to detailed knowledge about the specifics of the GP map. Surprisingly, we find that we can make such predictions quite successfully, or rather, we can bound such probabilities quite tightly. A key factor that allows these predictions is that we find phenotype complexity is important in determining phenotype probabilities. Thus, a broad class of GP map exhibits *simplicity bias*, where the structures of highest probability tend to

be simple, regular or symmetrical. While we are not the first to observe an inverse relation between complexity and probability in GP maps [88, 57], two key contributions of Chapter 4 are to extend such observations to many other cases, and (more significantly) to place these observations into the framework of *algorithmic information theory* [167, 106, 23, 24] and especially the *coding theorem* [110], and to use this framework to make quantitative predictions about some probabilistic properties of the maps. Lastly, although the work of Chapter 4 was motivated by studying GP maps, we stress that the results presented can be applied to a broad variety of input-output maps; indeed input-output maps appear in very many fields of mathematics, science and engineering. In this connection, for the mathematician or physicist, Chapter 4 may be the most significant and interesting.

Finally, we present some concluding thoughts to the thesis in Chapter 5.

Chapter 2

The Distribution of Genotypes Over Phenotypes

Genotype-phenotype (GP) maps commonly exhibit strong degeneracy, with many different genotypes generating the same phenotype. Given this excess of genotypes, it is interesting to ask: How are these genotypes distributed over the range of phenotypes? Are they roughly uniformly shared between phenotypes? Or, do some phenotypes account for many more or fewer genotypes than others? Further, do these distributions have any consequences for evolutionary dynamics and outcomes?

Here we review the literature and look for studies that directly or indirectly give insight on the distribution of genotypes per phenotype. We study GP maps from a wide range of biological contexts, and in both organismic and computational model settings. Strikingly, we find that many of these maps show dramatic variation in the number of genotypes per phenotype, and hence show 'bias' where a small range of phenotypes have disproportionately many genotypes underlying them. Further, we review theoretical analysis showing that this bias can have a pronounced effect on evolutionary dynamics; and, accordingly, we cite studies which indicate that phenotypes with disproportionately many genotypes associated tend to be overrepresented in nature.

2.1 Introduction

The fundamental distinction between *genotypes* (the genetic make-up of an organism) and *phenotypes* (the physical and physiological traits of an organism) [20] was

first recognised and highlighted by Wilhelm Johannsen in 1909 [31, 87, 86]. This classification emerged from Mendel's foundational work on heredity, which implied a separation between what was actually inherited, and what was subsequently manifested via development [112]. A common earlier theory was *preformationism*, where miniature organisms were thought to exist already fully formed in one of its parents [54]. In this view, organisms merely grew in size, without going through the now-known stages of genotypically orchestrated development. In the 1880s, August Weismann also distinguished between the body (*somoplasm*, or in a sense 'phenotype') and the material present in the fertilised egg (*germ-plasm*, or in a sense, the 'genotype') [112]. Nonetheless, it was Johannsen who coined and popularised the terms "genotype" and "phenotype" [31].

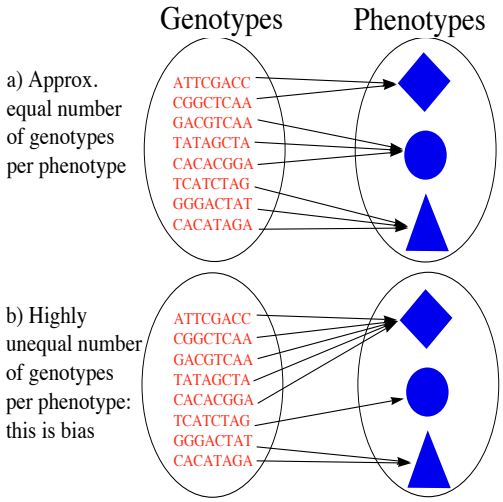


Figure 2.1: A cartoon illustrating bias in GP maps. Both diagrams have 8 genotypes mapping to 3 phenotypes (shapes). a) Each phenotype has approximately the same number of genotypes underlying it: 2 (diamond), 3 (circle) and 3 (triangle). b) The phenotypes have very different numbers of genotypes underlying them: 5 (diamond), 1 (circle) and 2 (triangle). *Phenotypic bias* [15] refers to the GP map's bias for certain phenotypes due to having disproportionately many genotypes underlying them.

In the century since the introduction of these terms, the study of the relationship between genotypes and phenotypes has benefitted greatly from technological advances, allowing unprecedented access to full organismic DNA sequences, detailed

investigations of developmental mechanisms, and computational simulations spanning biological length and timescales. In light of these advances, the *genotype-phenotype (GP) map* — i.e. the association of genotypes to phenotypes — is increasingly recognised as being essential to understanding biology and evolution [3, 141, 142, 121, 185, 120, 47].

Sewall Wright [193] recognised early on that Mendel's theory of particulate inheritance meant that the space or collection of possible genotypes underlying organisms is *vast*, resulting from the combinations of different alleles at different loci (Box 1). It follows then from Wright's observation, that for GP maps too the space of genotypes is astronomical; for example the number of different length 100 proteins is $20^{100} \sim 10^{130}$, which is a very large number, indeed 'hyperastronomically large' [95].

Box 1: The vastness of genotype spaces

Sewall Wright was among the first to comment on the huge size of genotype spaces resulting from simple combinatorics applied to collections of alleles or nucleotide bases. In 1932 he wrote [193]

> Estimates of the total number of genes in the cells of higher organisms range from 1000 up... With 10 allelomorphs in each of 1000 loci, the number of combinations is 10^{1000} which is a very large number. It has been estimated that the total number of electrons and protons in the whole universe is much less than 10^{100}.

Numbers such as 10^{1000} have been described by Kauffman as 'hyperastronomical' [95], as they are beyond even the kinds of numbers that are used in astronomy.

To make such numbers more tangible, we can ask: For what length RNA sequence would making all possible sequences take more mass than is available in the visible universe? The mass of the visible universe has been estimated at 10^{54}kg [108], and the mass of a single RNA nucleotide is about 10^{-9}pg [46] or 10^{-24}kg, which means we need to have $10^{54}/10^{-24} = 10^{78}$ nucleotides. For a length L RNA sequence, there are $L4^L$ nucleotides in all possible sequences, and we find that $L = 126$ gives $L4^L \approx 10^{78}$. Thus for even relatively short RNA sequences, making every possible sequence would require the mass of the visible universe, and many natural RNAs are much longer than 126 bases.

However, while genotype spaces in GP maps are vast, it is not true that only single sequences are associated to any given phenotype; instead, often there is great

degeneracy [49, 29] with very many sequences underlying the same structure or function [83, 176, 58, 182]. Indeed, while the 'neutralism-selectionism' debate still goes on [98, 77, 183], it is at least agreed that for a given phenotype such as a protein tertiary structure, there is redundancy in the GP map, with many genotypes *in principle* able to map to a given phenotype. (We stress 'in principle', as we are focussing on the biophysical question of which genotype sequences *could* — according to the laws of biochemistry — fold to a given structure; which is not the same as how many different sequences are found in nature with the same structure.) Now, this strong degeneracy means that if N_G denotes the total number of genotype sequences, and N_P the number of corresponding phenotypes, then we have: $N_G \gg N_P$. Given this observation, it is interesting to ask how this excessive number of genotypes is distributed between the phenotypes of a GP map: Are they evenly divided such that all phenotypes have the same number of genotypes? Is there only slight variation? Are there any biases in the mapping process that result in some phenotypes having many more genotypes than others?

At this point it is important to highlight some complicating factors when considering the quantitative distribution of how N_G genotype sequences are distributed between N_P phenotypes. Defining and counting the range of possible genotypes is relatively tractable, due to the discrete nature of genotypes. So, using simple combinatorics we can for example derive N_G, as we did when counting the number of protein sequences of length 100. In contrast, defining and counting phenotypes to derive N_P requires a little more thought: Firstly, phenotypes are not always discrete traits, but instead a phenotype may adopt a spectrum of traits (e.g. the colour of a animal's coat). In this case, it is difficult to enumerate distinct phenotypes to derive N_P and form a GP map. Secondly, even if a phenotype may be some discrete and countable trait, a given genotype may adopt a range of such phenotypes. As an example, if a number of RNA structures are within a few k_BT of the lowest free-energy ground state, then the RNA will fluctuate between different structures. More generally, there are many known cases of 'phenotypic plasticity' [143] where a single genotypes gives rise to different phenotypes within an organism depending on the environment, with

examples ranging from the rock ptarmigan changing its feather colours in the arctic spring [140], to multiple structures and/or functions for a given protein sequence [134, 177]. In these cases, deciding which characteristic to count as 'the' phenotype of a given genotype is problematic. Finally, while the common notion of a phenotype is that it represents the characteristics of an organism [150, 20], it is not always easy to decide which characteristics are relevant. From an evolutionary perspective, we may only consider a trait to be a phenotype (or at least an interesting phenotype) if it can be selected via imparting a non-trivial fitness difference. However, this may be too restrictive, as, say, a zoologist may be interested in some trait of an organism, even though the trait does not affect fitness (e.g. a 'spandrel' [70]). Finally, assigning a fitness effect to a given trait is usually not at all straightforward, hence exacerbating the difficulty of determining a phenotype.

Having discussed some challenges for general GP map studies, we will side-step these by examining only more tractable model GP maps where the phenotypes are well defined and permit a clear and quantitative relation of N_G genotypes to N_P phenotypes. Thus we can return to the biophysical question we posed above regarding quantitatively how genotypes are distributed between phenotypes. Somewhat surprisingly, the answer suggested by the GP map examples reviewed here is that a strongly non-uniform distribution of genotypes per phenotype is common: Within a given GP map, some phenotypes have proportionally very many genotypes underlying them, while some other phenotypes have proportionally very few. Another way to state this is that the maps exhibit a probabilistic 'bias' for certain phenotypes, in the sense that certain phenotypes are far more likely to be generated with a random choice of genotype (or random mutation) as compared to most other phenotypes[1]; see Figure 2.1. These examples of bias incorporate many aspects of biological organisation, such as bias in the universal genetic code, RNA secondary structure, protein enzyme function, a signalling circuit model, a neuron development model, and other

[1]We assume all genotypes are equally likely, which, while not usually the case due to e.g. mutation biases which are typically rather small, is a reasonable first-order modelling assumption. Further, given the strong biases observed, it would be very surprising if the effects of mutation biases etc. cancelled out the effects of the strong non-uniformity in distribution of genotypes per phenotype.

examples. In light of these many examples, we hypothesise that phenotypic bias is a general property of GP maps.

Observations of such bias in various GP maps has lead to various associated technical terms. Namely, the number of genotypes underlying a given phenotype has been called the phenotype's *designability* [114, 136]; *abundance* [36]; *degeneracy level* [15]; *genotype set size* [147]; *neutral network (NN) size* [160, 184], where the NN is the set of genotypes mapping to a given phenotype, and *neutral set size* (NSS or NS size) [156]. Of these names, we prefer to use NSS, but we will also use the term 'degeneracy' interchangeably. The phenomenon of a highly non-uniform distribution in NS sizes is called *phenotypic bias* [15].

Beyond the biophysical question of whether or not bias exists in given GP maps, we are also interested in the implications of bias in relation to evolution, especially the possibility that bias may provide some directional force in determining evolutionary outcomes. To give some context to this question, we recall that in the Modern Synthesis[2], the only significant cause of a particular direction in evolution is assumed to come from natural selection [142, 71], and this view is common today [13, 39, 84, 64]. Indeed, in the 1930's Fisher [61] and Haldane [78] argued that selection would typically overcome any biases or mutationally induced direction. This was inferred from a simple mathematical argument of balancing the probability of an allele being produced by mutation, with the probability that it is lost from the population via selection. As typically mutation rates are low and fitness effects are assumed to be relatively high, this 'mutation-selection balance' argument suggested that biases can be safely ignored as a directional factor in evolution. In addition, an even distribution of genotypes per phenotype is not an unreasonable assumption, as it is in fact the maximum entropy (i.e. most likely or 'best guess') distribution [162], given only the knowledge that N_G genotypes have been divided between N_P phenotypes. These factors are partially responsible for the common practice of ignoring the number of genotypes

[2]The *Modern Synthesis* is the name given to the body of theory developed largely in the 1930s and 1940s that brings together Darwinism, Mendelian genetics, and mathematical population genetics. Although many aspects of evolution theory are contested, it is commonly regarded that the Modern Synthesis represents the default and standard view of how evolution works.

per phenotype as a possibly important consideration in evolution; and, although not explicitly stated, a view which assumes that genotypes are evenly distributed between phenotypes, which we infer from the lack of attention given to this issue as a point of possible importance.

In this connection, it is significant that many of the works relating to phenotypic bias that we review below indicate that the bias has a pronounced influence on evolutionary dynamics and outcomes. Hence, despite the mutation-selection arguments, in light of the examples of bias and their effects on dynamics and in agreement with Maynard Smith *et al* [166] and others [70, 122, 191, 2] (see also below), we suggest that the role of (phenotypic bias) is currently undervalued in evolution, and that bias is important for evolutionary dynamics and outcomes. See Box 2 for a summary of our main hypothesese.

Box 2: Main hypotheses
The central hypotheses of this Chapter are that:

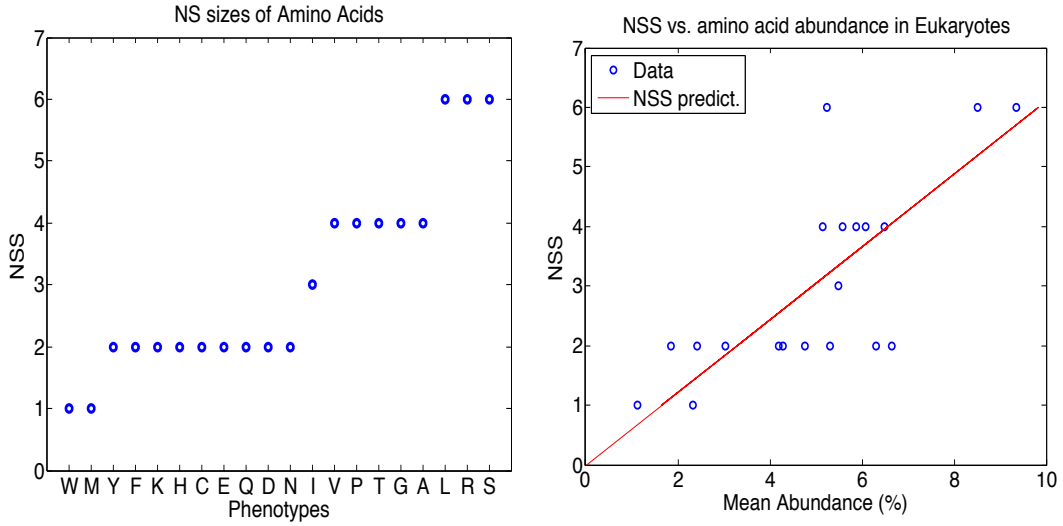
- For a given GP map, the space of all possible genotypes is distributed between phenotypes in a highly non-uniform manner: Some phenotypes have disproportionately many genotypes associated, and in this sense there is a bias for certain phenotypes. Such phenotypic bias is a general feature of GP maps, and is intrinsic to the mapping process.
- Phenotypic bias can affect evolutionary dynamics and outcomes, providing some directional force to phenotype evolution. Hence, high degeneracy phenotypes may be overrepresented in nature.

2.2 Examples of phenotypic bias

We now review several examples of phenotypic bias. The examples span biological levels from the genetic code to development, and as such we have organised these into sections: The genetic code, RNA secondary structure, proteins, biological networks, and development.

2.2.1 The genetic code

The genetic code is the association of the 64 codons (nucleotide triplets) to the 20 (natural) amino acids and the stop signal. In 1968 Francis Crick famously described the form of the genetic code as a 'frozen accident' [37], meaning that the specific association of codons to amino acids was merely a random event in life's early history, which had then become fixed in all descendant organisms. There were at least two good reasons for supposing this: Firstly, at the time there were no other known genetic codes. Secondly, Crick argued that for the code *itself* to evolve would be very difficult, as a small change in the code would likely have many phenotypic effects (i.e. pleiotropy) on an organism and hence be deleterious.



(a) (b)

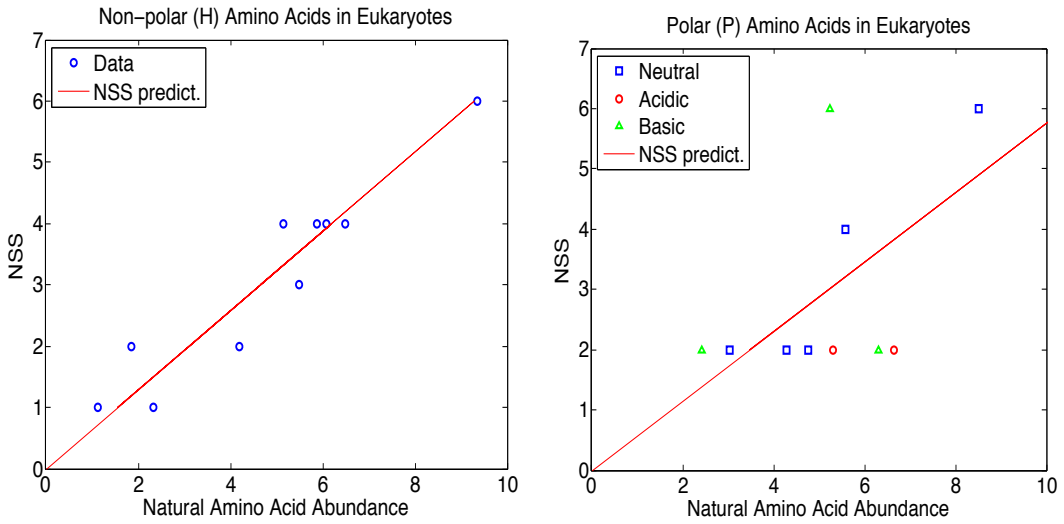
Figure 2.2: NSS distribution and NSS vs. natural eukaryotic abundance of amino acids. a) Non-uniform distribution in the number of codons per amino acid. There are 20 amino acid phenotypes, which we have ranked by NS size, and labeled with their standard one letter abbreviations. The amino acids with largest NS size have 6 codons mapping to them, with a gradual decay down to the two amino acids which have only 1 codon each. b) Amino acid NS size vs. % abundance in eukaryotes (average frequency of finding a particular codon in the coding regions of the genome, averaged over 5 genomes). The linear correlation coefficient is $R = 0.74$. The NSS prediction line shows the predicted abundances if all codons were equally likely. Data taken from [69].

Crick's view has since been challenged, partly on the grounds that in fact other genetic codes do exist [10], for example many mitochondrial genetic codes differ slightly (by 1 — 4 codon assignments) from the standard 'universal' genetic code, and even nuclear codes found, for example in some fungi and algae, have codes which differ slightly from the universal version [102]. These variations show at least that the code is not 'frozen'. Given also that some of these code alterations are beneficial, Knight *et al* [102] conclude that the genetic code is in fact still evolving. A separate consideration questions the random or accidental nature of the genetic code because the code appears to be highly optimised with respect to several measures [123, 65, 66, 69, 85], such as robustness to mutations.

Viewed from the GP map framework that our thesis focusses on, the genotypes are the different codons, and the phenotypes are amino acids, and the mapping is the process of translation, e.g. GCG $\rightarrow$ Alanine. As mentioned, there are 64 (4^3) possible codons, and of these 61 map to the 20 (natural) amino acids, and three are stop codons. It follows that the (universal) genetic code is degenerate with many amino acids mapped to by more than one codon. However, these 61 codons are not evenly shared between the amino acids, rather, the amino acids vary considerably in their NS sizes. For example, serine has 6 associated codons, isoleucine has 3 codons, and tryptophan has only 1 codon. That is, the NS size distribution is biased; see Figure 2.2(a). Thus there is phenotypic bias in the genetic code.

While this bias is not very strong, it nonetheless appears to have observable biological consequences: Mackay [123] and subsequently several others [100, 69, 85] have observed a good match between amino acid NS sizes and their abundance in nature (i.e. abundance in genomes, not abundance by gene expression). In Figure 2.2(b) we plot data (from [69]) for amino acid NS sizes and abundances for eukaryotes, averaged over proteins of five model organisms (*Arabidopsis thaliana, Caenorhabditis elegans, Drosophila melanogaster, Homo sapiens* and *Saccharomyces cerevisiae*). The plot shows a good positive linear relationship, with correlation coefficient of $R = 0.74$. Analogous calculations for bacteria and archae also show fair positive correlations of $R = 0.64$ and $R = 0.63$, respectively (data from [69]).

Noting that the central properties of amino acids (for protein structures) are their polarity (i.e. polar (P) or non-polar (H)) [43] and charge (i.e. neutral, acidic, basic), we looked to see how the correlation in Figure 2.2(b) changed by grouping amino acids of similar properties. We calculated the correlation between abundance and NS size



(a) (b)

Figure 2.3: Degeneracy and natural abundance of polar and non-polar amino acids. a) Non-polar (H) amino acid degeneracy (i.e. NS size) vs. natural abundance. The red line is an abundance prediction, which predicts abundance is equal to: (total % of 'H' amino acids)× (amino acid NS size)/61. The match of the data to the prediction is striking. The linear correlation coefficient is $R = 0.95$. (b) Polar (P) amino acid degeneracy vs. natural abundance, with degeneracy prediction line analogous to the above: (total % of 'P' amino acids)× (amino acid NS size)/61. The linear correlation is much less strong ($R = 0.52$). The different acidic natures of these polar amino acids have been plotted with different colours and symbols. Visually, the correlation of neutral amino acids (blue squares) is good. Abundance data taken from [69].

for non-polar (H) amino acids, and found it to be very strong, having a correlation of $R = 0.95$; see Figure 2.3(a). If selection had no way to distinguish between H amino acids, then we would expect (by neutral mutation) to have a very strong correlation between degeneracy and abundance. However, even without making the contentious assumption of prevalent neutral mutations, other factors may favour the abundance of high degeneracy amino acids (see discussion below). Performing the

same analysis for all polar (P) amino acids together, the correlation is less at 0.52; see Figure 2.3(b). Additionally, for the polar variety we have distinguished the acidic properties in the plot with different colours and symbols. Note that amino acids can be distinguished by many different properties, and their acidic character is one of these standard classifications, and expected to be an important one. It appears that the neutral amino acids correlate better than the charged ones, but it is not clear to us why the correlations in the cases of polar and non-polar should be markedly different. The reason may be that as polar amino acids are more likely to be on the surface of a protein, they are more likely to be subjected to selective pressures.

If mutations were truly random, then the percentage of GC pairs and AT pairs would each be 50%. However, in genomes these fractions can vary. We therefore briefly examined the effects of varying these fractions on the overall correlations shown in Figure 2.3. We found that they do not strongly affect the overall correlation between NSS and frequency.

Having examined natural abundances, an interesting and central question is: Has natural selection shaped the genetic code so that the amino acid degeneracy patterns match that amino acids requirements of organisms? Or did the degeneracy pattern cause the abundance pattern? Or is it some combination or other process? Mackay seems to have assumed the former, but he did not provide any clear supporting evidence for this position, though the code's ability to evolve (above) may support this view. On the other hand, in 1969 King and Jukes [100] argued (in a foundational paper of the neutral theory of evolution) that the correlation is a result of neutral evolution, that amino acid frequencies are close to those expected by roughly uniform random mutation. They pointed out that under strong selection there would be no reason for any strong correlation between amino acid degeneracy and abundance in nature, as presumably only amino acid fitness effects (related to physico-chemical properties) would matter. On the other hand, if there was no selection at all, a very strong correlation between degeneracy and natural abundance would be expected, as random mutations would make codons roughly equally likely, and so an amino acid's abundance would be proportional to the number of codons associated to it. One

might argue that the presence of mutational bias, especially strong GC content bias will also affect the assumption of equally likely codons. However, we have checked that in general this doesn't strongly change the bias in amino acid content, and even if it did, one could simply adjust the prediction of what a null model of only neutral mutations would generate (see also discussion below).

Richmond [149] responded critically to King and Jukes' paper. In reference to amino acid abundance patterns, his main objection was that if neutral evolution was responsible for the correlation, then we would expect to see correlation of relative amino acid mutability (mutation rates between amino acids), which was not seen in the small data sets available in 1970. Recently however, Stoltzfus and Yampolsky [171] have shown that much of the chimp-human amino acid divergence pattern can be explained by invoking various mutational biases, with the degeneracy pattern of the genetic code being a significant factor in this. This contributes to addressing Richmond's objection. More recently, for example Gilis *et al* [69] have also discussed this issue of the direction of causality between degeneracy and abundance, but did not commit to either view, suggesting also that it may be some combination of factors and co-evolutionary process.

We highlight that the genetic code appeared very early in life's history, and extant versions differ only very slightly. As such it would seem strange — if the code adapted to very early prokaryote organism requirements — that such similar amino acid requirements would apply to such a broad range of organisms, including eukaryotes, that inhabit different niches and are subject to different selection pressures and strengths. Further, we stress that although a neutral theory of evolution [98] would explain the observation in question, we do not need to commit to a position on the neutralism-selectionism debate, as other explanations may serve equally well. For example, it could be that there is a selective advantage that several amino acids fulfil. In that case, whichever amino acid has the largest NSS is the most likely to be presented as variation and fix. Once it fixes, then the selection pressure goes away. In this way the NSS could lead to biases, even when there is selection. As an example of such a selective advantage, consider a protein that needs to become more stable.

One way of doing this may be to change a polar amino acid to a hydrophobic one. A hydrophobic amino acid like leucine, with 6 codons mapping to it, will be much more likely to appear than a hydrophobic one like methionine, which only has one codon mapping to it.

Germane to the preceding discussion, Li [118] (and others, e.g. Ref. [75]) have shown that GC content bias, arising from mutational bias, strongly correlates with, and appears to affect, amino acid abundance in bacteria. Specifically, amino acids with codons of high (low) GC content were more (less) abundant when genome GC content increased from mutational bias (as opposed to selection). This is relevant because if GC mutational bias affects amino acid abundance, it is more plausible that the mutational bias resulting from varying amino acid NS sizes also can affect amino acid abundances. Knight *et al* [103] also studied the abundance of amino acids and GC content, and emphasised that the causality flow should be from GC bias to amino acid abundance patterns, not the converse direction. They argue that their ability to predict aspects of amino acid abundance (in a broad range of organisms) from GC content alone would be very surprising if in fact there were specific selection pressures for amino acid abundances. Additionally, a good positive correlation was found between the GC content of intron, exon, noncoding, tRNA and rRNA sequences within bacteria [132], which Knight *et al* [103] suggest implies that common forces act on the different parts of the genome; i.e. mutational bias are causing the GC content bias, rather than selection for specific amino acid abundances. Hence, they hold that it does not seem likely that the correlation of GC content and abundance results from the genetic code adapting to selection for a specific abundance pattern of amino acids.

In summary, while it is presently not certain that amino acid NS sizes influences natural amino acid abundances or if the code has adapted to the abundance requirements, it seems that the former view is more plausible. So, the genetic code exhibits phenotypic bias and this bias, while not being extremely strong, appears to affect the frequency of amino acids in nature.

Next we turn to systems where bias is much more pronounced.

2.2.2 RNA secondary structure

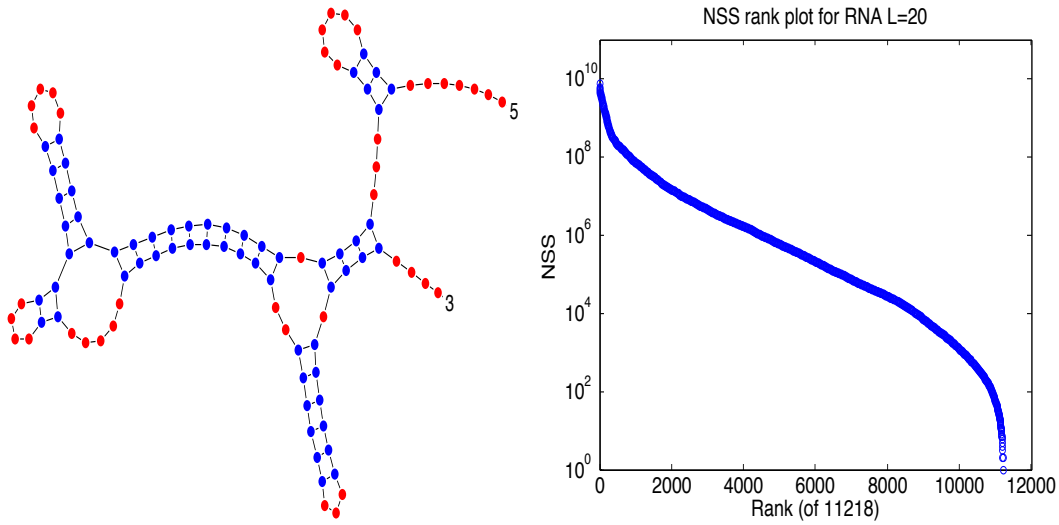
One of the best studied model GP maps examined in the literature is that of an RNA nucleotide sequence (genotype) folding to its minimum free energy secondary structure (SS) (phenotype); see Figure 2.4(a). RNA SS folding has been a popular choice because it is computationally reasonably tractable (unlike protein folding), and the necessary software is widely available thanks to software such as The Vienna RNA Package of Hofacker *et al* [81], introduced in the early 1990's.

Although the most important phenotype of an RNA may be said to be its function, the SS is essential to the function, and hence worthy of analysis itself. For example, correct SS is essential for many viruses to interact with proteins, and also for messenger RNA to efficiently translate into proteins [182]. A similar message is currently emerging from studies which show that RNA global conformations and the structural conformations adopted in response to physiological signals, are strongly constrained by (and hence largely defined by) the SS [7, 8]. For these reasons, RNA folding to SS remains a biologically realistic and relevant GP mapping [160, 201, 62]

For an RNA nucleotide sequence of length L, and 4 possible nucleotides per base, there are 4^L different genotype sequences. The number of associated phenotypes also grows exponentially, but much more slowly [74], approximately proportional to 1.76^L (see next chapter). Hence we see that the map is strongly degenerate, with many more sequences than associated structures.

Due to the exponentially increasing size of the genotype and phenotype space for this GP map, complete enumeration of sequences is computationally demanding for large L. However, recently Schaper and Louis [156] performed such a complete enumeration for $L = 20$, so we use their data to illustrate the quantitative properties of the GP map: There are $N_G = 4^{20} \approx 10^{12}$ genotypes but only $N_P = 11218$ different SS (phenotypes), excluding the trivial structure which has no bonds.

It has long been known from folding simulations that the RNA GP map shows phenotypic bias [160], and we illustrate this bias with the data for full enumeration for $L = 20$, see Figure 2.4(b).



(a) (b)

Figure 2.4: RNA secondary structure and NSS. (a) An example RNA secondary structure: A random length $L = 100$ RNA nucleotide sequence (genotype) was generated and computationally folded to its minimum free energy structure (phenotype). Blue dots are bonded bases, and red dots are unbonded bases. b) NSS plotted against rank for RNA $L = 20$. There are $N_P = 11218$ different RNA secondary structures, excluding the trivial structure. Note the log scale. Some structures have $\sim 10^{10}$ sequences associated and some only ~ 1, with a progressive decay in between. Data generated by Schaper and Louis [156].

Box 3: Quantifying bias

How can we quantify phenotypic bias in GP maps? The essential feature to measure is the degree of variation in the number of genotypes per phenotype. One simple approach is to use the standard deviation (σ) of NS sizes divided by the mean (μ) NS size. For the genetic code this yields $\sigma/\mu = 0.51$; and for RNA with $L = 15$ we have $\sigma/\mu = 2.26$, so the variation is greater — hence the bias is stronger — in RNA than the genetic code.

Another approach is to use the concept of entropy from statistical physics and information theory. The Shannon entropy H (in bits) of the distribution of NS sizes is

$$H = -\sum_{i=1}^{N_P} \frac{\eta(q_i)}{N_G} \log_2 \left(\frac{\eta(q_i)}{N_G} \right)$$

where $q_1, ..., q_{N_P}$ are the N_P different phenotypes, $\eta(q_i)$ is the NS size of phenotype q_i, and N_G is the total number of genotypes. If all the NS sizes are equal, then $H = \log_2(N_P)$, and if one phenotype has a very large NS and the rest very small, then $H \approx 0$. More generally, entropy corresponds to log of the number of states (or number of phenotypes). As a result, the exponential of the entropy 2^H is used in statistical physics and information theory [128] as a rough measure of the effective number of states, or in this case the effective number of phenotypes. For example, in the case of all equal NS sizes, we have $2^H = 2^{\log_2(N_P))} = N_P$, reflecting the fact that all phenotypes are well represented in terms of NS size; and for the case where one phenotype has nearly all the genotypes we have $2^H \approx 2^0 = 1$. As 2^H gives the effective number of phenotypes, we can define a *bias ratio* $\beta \in (0,1]$

$$\beta = \frac{2^H}{N_P}$$

which is a number giving the ratio of the effective number of phenotypes to the total number of phenotypes. Hence β can be used to compare the strength of bias in different GP maps: If $\beta \approx 1$ then the genotypes are fairly equally distributed between phenotypes, and if $\beta \ll 1$ then the majority of the genotypes map to a tiny fraction of phenotypes (i.e. the bias is very strong).

Because the majority of all the genotypes map to a fraction β of all phenotypes, we prefer this measure over σ/μ, for which it is harder to see direct links to GP maps. By way of example, for $L = 20$, the majority (88%) folding genotypes map to a fraction of $\beta = 0.06$ of structures. This shows quantitatively that the bias is strong for the $L = 20$.

We note that this definition is very general, it can therefore be applied to a wide range of different kinds of GP maps.

Notice that the distribution is strongly non-uniform, with the highest ranked phenotypes having $\sim 10^{10}$ genotypes mapping to them (while the trivial phenotype with no bonds has $\sim 10^{12}$ genotypes), then the degeneracy decays until one structure (rank

11218) has only 1 genotype sequences mapping to it.

Phenotypic bias is not merely an artefact of short RNA sequences, as qualitatively similar plots arise for longer sequences as we shall see in the next chapter. Also, other authors have partially sampled longer RNAs, also finding bias; see e.g. [160, 159, 91]. However, this sampling is only very partial because of the exponential growth of the spaces.

Phenotypes with NS sizes larger than the mean NS size are known as *frequent structures* [182]. From simulations, it has been observed that as the length L increases, the proportion of phenotypes which are frequent *decreases*, while the proportion of genotypes that map to frequent phenotypes *increases* [182]. Consequently it is believed [182] that as L grows very large, nearly all the genotypes will map to a vanishing fraction of phenotypes (i.e. the 'frequent' ones). We return to study this in detail in the next chapter.

At this point, we introduce a new quantitative measure of bias, as a means to quantify the strength of non-uniformity in NSS distributions. In Box 3 we define a quantity β which is proportional to the 'effective' number of phenotypes in a GP map which collectively account for the majority of all genotypes in a map.

Hence β can be used to measure the strength of the bias: If the effective number of phenotypes is only a small fraction of all phenotypes ($\beta \ll 1$) then the bias is strong; but if instead the effective number of phenotypes is roughly the same as the total number ($\beta \approx 1$) then the bias is very weak. The form of β comes from statistical physics and information theory [128]. As an example, for the RNA $L = 20$ NSS data displayed above, we find that $\beta = 0.06$, so the bias is strong. We will use this measure (as well as the more common standard deviation over the mean measure) for many of the maps that follow.

In the case of the genetic code, we examined the relation between degeneracy and natural abundance. Hence a natural question to ask in the context of RNA SS is: Where on a NS size vs. NS size rank plot would we find *natural* RNA secondary structures? *A priori* it is not clear, though one may reason that natural RNA structures, as a result of selection, may be complex and finely sculpted forms, and thus have a

low rank (i.e. small NS size). However, an important study [91] which estimated NS sizes for 82 naturally occurring RNA secondary structures in the fRNAdb database [99] with $30 \leq L \leq 50$ found the opposite: The NS sizes of these RNAs structures were larger than 99.99% of randomly chosen structures. This means that for this study, the phenotypes found in nature were those of largest NS sizes. fRNAdb is a non-coding RNA sequence database, and the sequences used came from a variety of organisms (humans, fruit flies, thermophile archaea, arabidopsis plants, yeast etc.) as well as several different functions (snRNA, gRNA, ribozyme etc.) so it is unlikely that sampling bias is responsible for this striking observation. A similar conclusion was drawn by different authors [36] using a heuristic method for estimating phenotype NS sizes, and while using Rfam, a curated database of functional RNA genes. They concluded that the SS in the database tend to be those of larger NSS; specifically, nearly all types of RNA molecules examined were ranked close to the 70 percentile mark, meaning that roughly 70% of randomly chosen structures would have smaller NS sizes. In the next chapter we extend these studies of natural RNA data, and find the effect of NSS is more striking even than these studies suggest.

Finally, we speculate on bias in RNA global tertiary structure. As strong bias is observed in the sequence-secondary structure map, and there is evidence [7, 8] that this structure plays an important role in defining the dynamic aspects of RNA global structures, it is probable that the sequence to RNA global structure map will also show strong phenotypic bias.

2.2.3 Protein structure and function

2.2.3.1 Protein tertiary structure

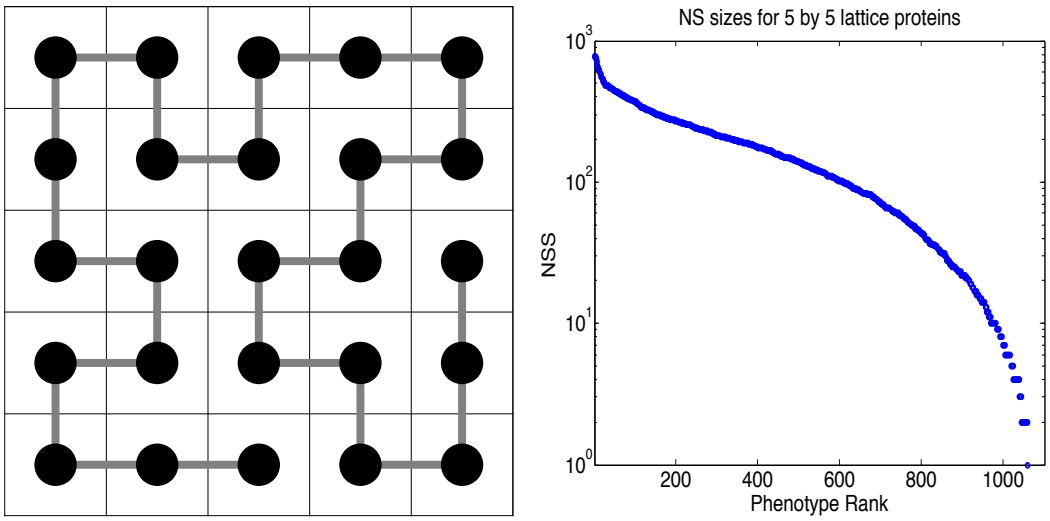
Among the most familiar GP maps is that of a sequence of amino acids (genotype) folding to its minimum free energy protein tertiary structure (phenotype). The fundamental role these biomolecules play in living systems means that they are a focus of much research. Analogously to RNA structures, one may consider the most important property of a protein to be its function rather than simply the structure;

nonetheless, the tertiary structure is essential for function [126], and hence worthy of independent study.

A characteristic feature of the protein folding GP map is that the majority of sequences do not fold stably, so do not have a well defined tertiary structure phenotype [96, 174], though the extent of this 'majority' remains unresolved [96, 174, 6, 28]. Despite this, experimental studies show the high tolerance of proteins to mutations [76], and homologous proteins can show strong sequence divergence [125, 29, 48]. So there is degeneracy in the mapping, but is there phenotypic bias? To answer this we must be able to estimate the NS sizes of protein tertiary structures; how can this be done? Using natural sequence divergence may give an indication, but the connection between natural sequence divergence and NS size is not necessarily direct, as epstasis and the connectivity of sequences affects mutational exploration of a phenotype's sequences [155, 12, 11, 188]. Could we employ a direct experimental approach? Not practically: Experimental analysis of NS sizes would be extremely difficult due to the huge number of sequences required; indeed complete chemical synthesis of all sequences is impossible even for modest sized proteins of length 100, for which there are $\sim 10^{130}$ different sequences. Even sampling a significant fraction of sequences would be financially taxing and labour intensive. Hence, in order to make progress in exploring possible phenotypic bias in proteins, we will look at both a simplified computational model of protein folding, and analytic approaches to the problem of protein degeneracy, or 'designability' [114].

We begin with an extensively studied model, known as the HP lattice protein model [196]. In this framework, only two amino acids constitute the protein sequence, one hydrophobic (H) and the other polar (P), rather than the 20 amino acids used in nature. These sequences of H or P amino acids fold into minimum energy structures that are confined to a 2D or 3D lattice. The model is motivated by the observation that a major driving force of protein folding is the hydrophobic force [44], which tends to make the polymer fold into a compact shape with a hydrophobic core. In the model, a sequence is said to be foldable if it has a unique lowest energy structure. Otherwise, it is said not to fold. Energies are calculated by counting the number of H-H, H-P, and

P-P bonds on non-sequential neighbouring residues. Despite its simplicity, HP lattice protein models reproduces some properties of real protein folding. Specifically, in both the model and real proteins, only a small fraction of sequences fold successfully [114, 6], and HP lattice proteins tend to adopt regular and symmetrical structures, just as real proteins do [114], as well as HP proteins exhibiting protein like secondary structures.



(a) (b)

Figure 2.5: The 5 by 5 HP lattice protein. a) An example structure for the 5 by 5 HP lattice protein. Contacts are formed between non-sequential adjacent residues. There are $2^{25} \approx 3 \times 10^7$ possible HP protein sequence genotypes, and $\sim 10^3$ different possible structures (phenotypes). b) A NS size-rank plot for a length 25 chain confined to a fold in a 5 by 5 square. The distribution shows bias with $\beta = 0.69$ and $\sigma/\mu = 0.89$. Data generated by Yang *et al* [195].

To illustrate the model, a 5 by 5 lattice protein is shown in Figure 2.5(a). Each possible structure in the square 2D lattice has the same number of contacts, but the energies of these structures differ, depending on the identity of the bonds (in this case the number of H-H bonds, because H-P and P-P are taking to have a zero contribution to fold energy). The NSS of a given structure is the number of HP sequences that fold *uniquely* to that structure.

In an important computation study of HP lattice proteins, Li *et al* [114]] made the observation that the NS size distribution for the structures was highly non-uniform (i.e. phenotypic bias): For the 3 by 3 by 3 cubic lattice (with 51704 structures), the number of sequences per structure ranged from ∼4000 progressively down to only ∼1, and qualitatively similar results were obtained for the other lattice simulations. Of the 2^{27}∼10^8 different possible HP sequences, 95.25% did not fold uniquely, and so were not included in NS size calculations.

The HP lattice model has been studied by very many others also, and the bias has been found to be robust to many model alterations such as using different bond energies [16, 115, 194]; using 20 amino acids with the Miyazawa-Jernigan matrix [115]; not confining the lattice folds to any particular global shape [16, 17]; confining the folds to shapes other than squares and cubes [139]; and including energies of local interactions between residues [195]. As an example, we give in Figure 2.5(b) the NS size-rank plot for a 5 by 5 HP lattice protein simulation (data generated by Yang *et al* [195]). The plot shows bias with some NS sizes of nearly 1000 and some only about 1; quantitatively, $\beta = 0.69$ and $\sigma/\mu = 0.89$.

We next turn to analytical treatments of protein degeneracy, and in particular study two approaches. Firstly, England and Shakhnovich [52] used a statistical physics approach to analytically derive a correlate to protein degeneracy. They argued that the main determinant of a fold's NS size is the fraction of its sequences which lie below a given energy cut-off, and that practically this translates to a large 'contact density' (CD), which depends on the tertiary structure geometry, and can be easily calculated from a protein's contact map. Shakhnovich *et al* [163] used the CD to investigate degeneracy of natural proteins from a database and compared them to the mean number of non-redundant sequences in a database for gene families. They found a very strong linear correlation (R^2 >0.9) between the logarithm of this mean and the CDs of the representative structures. Later, Ferrada and Wagner [57] performed a very similar analysis on protein domains, and found again that the number of non-redundant sequences per domain correlated strongly with CD (Spearman's rank cor., $r = 0.88$).

Secondly, Coluzza *et al* [34] have analytically derived an expression for a protein structure's degeneracy using the coarse-grained "Caterpillar" model of protein structures [33]. Their work shows that the degeneracy depends exponentially on the number of hydrogen bonds. As the number of bonds varies between structures, then this implies that exponential variation in NS sizes can be expected for natural proteins.

In summary then, there appears to be good evidence that protein tertiary structures exhibit phenotype bias. Might this bias affect evolutionary trajectories and outcomes as we saw for RNA? The works just cited using the contact density indicate that phenotypic bias in protein tertiary structures has tangible implication for natural protein evolution. Addressing this question earlier, Taverna & Goldstein [173] investigated the relationship between protein NS sizes and evolutionary outcomes. They used a 5 by 5 HP lattice protein model and simulated the evolution of these proteins, with the only fitness condition imposed on a sequence being that it fold stably. Their conclusion was that "population effects cause highly designable structures to be even more overrepresented". They attributed this overrepresentation of 'highly designable' folds to the fact that they are more robust to mutations.Finally, a study by England *et al* [51] looking at proteins in bacterial genomes found another connection to degeneracy: Thermophilic adaptations of bacteria often proceeded by deleting domains with smaller NS sizes and using orthologous proteins with larger NS sizes instead. They concluded that there was evidence for high degeneracy being "a key component of protein fitness", due to high degeneracy structures having greater thermal stability.

Nevertheless, on balance the question of whether or how phenotype bias affects protein tertiary structure in nature remains a fairly open question. The main reasons for this are firstly that the spaces are so huge, they grow as 20^L, and secondly that the protein folding problem is very difficult, so that something analogous to solving for large sets of genotypes, as done for RNA is not feasible. Instead, one has to resort to highly simplified models like the HP model. These do show strong bias, but the link to biological sequences is much harder to make than it is, for example, for RNA.

2.2.3.2 Protein enzymatic function

While the amino acid-protein structure relation has been studied extensively and several studies of NS sizes have been carried out, the presence of phenotypic bias in protein function is not so well studied. Having said that, several works are relevant: A mutation study of the lysozyme of bacteriophage T4 [148] found that 84% of mutations did not drastically effect function. Similarly, of 1634 single amino acid substitutions made in the *E. Coli lac* repressor, 55% were found not to cause loss of function [101]. These and other studies show that protein sequences can diverge strongly, while preserving original function [182]. It follows that the number of sequences underlying such degenerate functions must be very large indeed (i.e. the NS sizes), due to the many possible combinations of compatible amino acids [48]. Other experimental work appears to support this conclusion also, in that estimates of the probability of generating a given function from a random sequence of amino acids range drastically over many orders of magnitude from 10^{-11} for an ATP binding protein [96], to 10^{-23} for a helical bundel chorismate mutase [174], and as low as only 10^{-77} for *any* functional protein [6][3].

In a bioinformatics study closely relating to phenotypic bias in protein function, Ferrada and Wagner [58] examined enzymes with known structure and function with the aim of understanding how protein functions are distributed in amino acid sequence space. The UniProtKB/Swiss-Prot database was used, which is the reviewed and manually annotated subsection of UniProt (a comprehensive resource of protein sequence and functional information). They only studied single domain proteins of length <50, and also discarded sequences which had >99% similarity, as well as discarding any labeled "putative" or "by homology". This left 39529 sequences, which fold to 457 different structures (according to the CATH classification [73]), and constitute 1343 different enzymatic functions (defined in accordance with the Enzyme

[3]Clearly the latter estimate is not consistent with former two, but as Axe [6] has pointed out, there are at least two main methods for such estimations (either generating purely random sequences or testing the maintenance of function after making mutations) which undermines the ability to compare these.

Commission (EC) [9]). The authors observed that it is not possible to simply extrapolate from our knowledge of tertiary structures to actual protein function, as many proteins with the same structure perform different functions [133]. On the other hand, some functions can be performed by proteins with different structures, e.g. DNA polymerases [169].

One of Ferrada and Wagner's main results is that there appears to be phenotypic bias in the protein structure-to-function map. They found that most functions can be performed by only one structure, some functions can be performed by several structures, and a few can be performed by as many as 14 structures. Additionally, they plot (see their Figure S1(b)) the number of sequences (genotypes) per function (phenotype), which also shows bias with progressive decay: For the data they used, the number of sequences underlying different functions varies, with most functions associated with less than 25 sequences, and some with as many as 400 sequences.

These findings (and the above experimental studies) suggest that phenotypic bias may be present in the protein sequence-function map. However more investigation is required to firmly establish this, and Ferrada and Wagner caution that their "evidence has to be taken with a grain of salt." This is because of various sources of bias in the database, and the relatively small sample size of $\sim$40000 sequences. As an example, they note that certain enzymes may be better studied due to their medical value, and hence possibly be over represented in the database. Also they only looked at short (length < 50) single domain proteins, which may have different properties to general proteins. While these points may affect the quantitative details of the bias observed, we do not think these issues are sufficiently problematic to qualitatively change the conclusion that phenotypic bias is present in the sequence-to-enzymatic function map.

2.2.3.3 Protein quaternary structure

Above, we surveyed evidence for phenotypic bias in protein tertiary structure and function. However, in nature many proteins are composed of multiple polypeptide

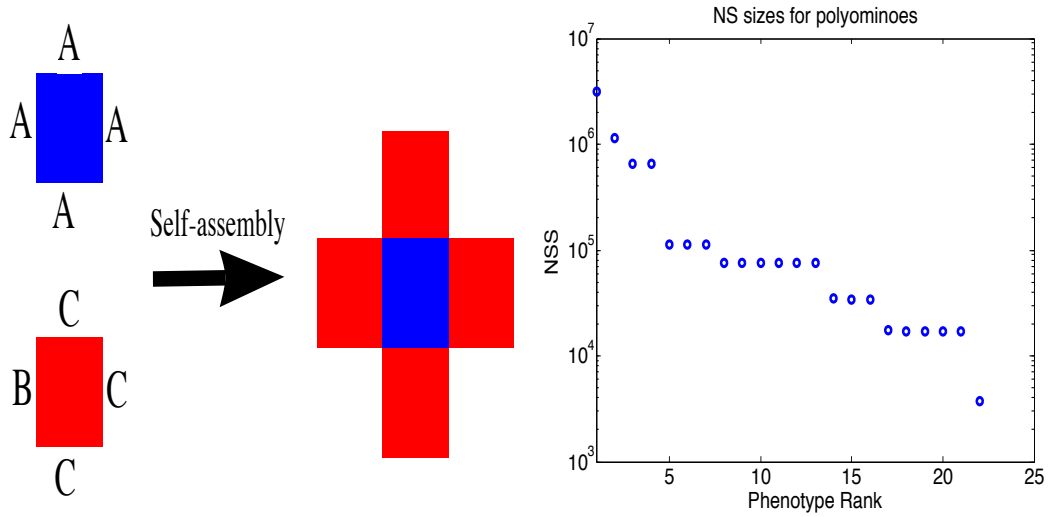
chains; these protein quaternary structures are known as oligomers, or protein complexes [131, 126]. The importance of these complexes has been highlighted by the research of the previous two decades, which has shown that interactions between proteins are at the heart of most biological processes [179, 175].

Just as with tertiary structures, protein complexes show large degeneracy in the sequence to quaternary map: A bioinformatics study found that more than 70% of homomers (oligomers of identical subunits) maintain quaternary structure over evolutionary timescales while sequences can have as little as 30% identity [111]. Interestingly, while there can be great sequence redundancy, protein interactions evolve faster than tertiary structures [175], and small changes in sequences can lead to large changes in quaternary structures [68, 164].

Regarding the presence of phenotypic bias in the sequence-to-quaternary structure GP map, direct experimental analysis faces the same problems as for tertiary structures, i.e. the number of possible sequences to synthesize and resultant structures to analyse is vast. However, as with the case of lattice proteins, we may get some insight from the behaviour of simplified models. Fortunately, a GP map 'polyomino' model of self-assembling protein complexes has recently been introduced [1, 90].

The polyomino model is as follows. Square tiles self-assemble — via interactions between the tile edges — into planar shapes known as polyominoes. Each tile is assigned a letter (or colour) to each of its four edges, and certain letters (or colours) attract each other. A tile is entirely specified by a binary genotype which determines which letters (or colours) appear on its four edges, and which letter (or colour) interacts with another is pre-set and fixed; see Figure 2.6(a).

Johnston *et al* [90] observed that this GP map exhibits strong phenotypic bias, and we plot here the data given in their paper, for a polyomino model which had 8 tile types, and 2 types of colours, depicted as the space $S_{2,8}$; see Figure 2.6(b). The NS sizes of the 22 phenotypes range from $\sim 10^6$ progressively down to $\sim 10^3$, and in this case $\beta = 0.28$ while $\sigma/\mu = 2.33$; this is very strong bias. Note that 61% of the genotypes did not deterministically generate a bounded polyomino, and so were discarded from NS size calculations.



(a) (b)

Figure 2.6: Polyomino example and NS size distribution. a) The binary string genotype specifies the letters on the edges of the two tiles, and the tiles self-assemble to form a structure (phenotype). As an example, if the block types are A,A,A,A and B,C,C,C with the specification that colours A and B attract, while colour C attracts no other colours, then the 'cross' polyomino (phenotype) is produced. By varying the colours on the tiles edges, and also the number of tiles, many different shapes (phenotypes) can be produced. b) The biased NSS distribution for the polyomino model of protein quaternary structure. The NS sizes of the 22 phenotypes range from $\sim 10^6$ progressively down to $\sim 10^3$; $\beta = 0.28$ and $\sigma/\mu = 2.3$ so the distribution is strongly biased. Data taken from [90].

The evolutionary dynamics of these self-assembling polyominoes were also studied, via simulations [90]. Various scenarios were considered such as changing mutation rate. They point out that NS sizes will likely affect evolutionary outcomes, due to the entropic (i.e. probabilistic) drive to phenotypes of larger NS size. Further, they point out that when a given functional requirement can be carried out by many different phenotypes, the ones of larger NS size will most likely be 'found' first. Illustrating this, they evolved the structures while selecting for a given size, and found that while there were many different structures of that size, evolutionary searches found the structures of larger NS size much more quickly than smaller NS sizes. Additionally, at high mutation rates, they found that the larger "mutational entropy" (i.e. NS size) of some phenotypes meant that the population's mean fitness was low, due to sub-optimal phenotypes of larger NS size winning out. Finally, they used their polyomino evolution study to rationalise the well known prominence of certain geometrical symmetries in real protein complexes.

The authors attribute the intrinsic phenotypic bias in these polyominos to the different amounts of information required to specify bonds for different phenotypes. As an example, the structure which is just a single tile only requires designing a block to have no bonds at all, which is relatively unconstraining on the genotype compared to (say) a large multi-subunit block phenotype which requires many bonds to be specified. While the polyomino model is very abstract, this explanation of bias in terms of genetic information constraint is not system specific, and may well apply to other GP maps.

In sum, we do not have direct evidence of phenotypic bias in protein quaternary structure, but in a simplified model of such structures very strong bias is observed. Further, in computer simulations of evolutionary dynamics, the bias was observed to significantly influence evolutionary dynamics and outcomes.

2.2.4 Biological networks

There has been great interest in biological networks in recent years due to a shifting focus in the biosciences from a molecular and gene-centric view to a multi-scale

interaction-centric view of living systems [5, 135]. Gene regulatory networks (GRNs) in particular have received much study because it has been realised that an important (if not central) process in evolution is changing gene regulatory interactions [38], and even relatively few mutations in regulatory regions can have large phenotypic effects [22]. For example, Stern and Orgogozo [170] have recently given analysis which shows that for longer term macroevolution, most mutations inducing morphological change are found in the $cis-$regulatory regions, whereas for short term microevolution within species, most mutations inducing morphological change are found in the protein coding regions. This study highlights the importance of understanding GRNs for understanding large scale morphological evolutionary change.

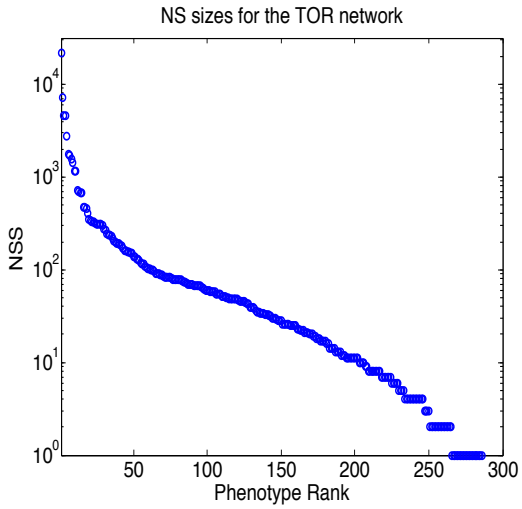


Figure 2.7: The NS size-rank plot for the TOR signalling circuit. There is very strong bias in this map, $\beta = 0.09$ and $\sigma/\mu = 5.9$. The NS sizes range from $\sim 10^4$ down to ~ 1, over 286 phenotypes. Data from Ref. [146].

A common simplification employed in modelling GRNs is the use of Boolean networks, pioneered by Kauffman [93], which coarsegrains the system via two assumptions. Firstly, it is taken that gene states are the variables under evolution, and not the concentrations of the gene products. Further these gene states are taken to be either "on" or "off", which is based on the observation of gene states being cooperative transitions, hence sigmoidal, which may be approximated by discrete states [94].

Secondly, it is taken that updating gene states is synchronous throughout the network. Despite these simplifications, Boolean models have been successfully applied to analysing the yeast cell cycle network [113], the segment polarity gene network of *Drosophila melanogaster* [4], the floral cell fate of *Arabidopsis thaliana* [53], and the GRN associated to endomesoderm specification in the sea urchin embryo [138]. This suggests that Boolean models capture many qualitative properties of biological networks, and so if there is bias in the models then it is probably also a feature of the true biological system.

2.2.4.1 Target of rapamycin signalling circuit

We begin with the work of Raman and Wagner [146] which looks at a systems biology (coupled differential equation) model of the target-of-rapamycin (TOR) signalling circuit for budding yeast (*S. cerevisiae*). TOR is a highly conserved protein kinase which controls growth in yeast, fly and mammalian cells. The aim of their study was to see how the topology of the circuit interactions affects the signalling circuit's behaviour.

Despite effort, uncertainty remains about the interactions between the components in the TOR network (i.e. the network topology). This pattern of molecular interactions is ultimately specified by the yeast's genome, and hence the authors defined the genotype in their model to be a binary string (of length 18) representing the topology of the interactions in the circuit. This yields $2^{18} \approx 3 \times 10^5$ different genotypes, but many of these were eliminated due to being biochemically incompatible, leaving 6.9×10^4 genotypes.

The circuit's phenotypes were determined on the basis of the concentration-time trajectories of eight key proteins complexes (e.g. Tor12). Specifically, the model generates a continuous concentration-time profile for each protein, and these were discretised by recording only the concentrations at a fixed number of time points. Subsequently, these trajectories were clustered into similar signalling behaviours using BIRCH (balanced iterative reducing and clustering using hierarchies) [200], which is a well known clustering algorithm used for large data sets. The algorithm clustered

the (coarse grained) signalling behaviours of key signalling molecules into 286 qualitatively different behaviours, which were then taken as 286 different phenotypes.

This GP map shows very strong bias with $\beta = 0.09$ and $\sigma/\mu = 5.9$; see Figure 2.7. Indeed, the largest genotype set contains 21633 genotypes, or 31% of the total $\sim$70000 genotypes, and only 26 (=286β) phenotypes account for nearly all (82%) of the genotypes. Very interestingly, of the 286 phenotypes in the GP map, the one most similar to the experimentally determined reference TOR signalling phenotype found in the yeast organism, is the phenotype with the largest NS size. (This observation is similar to that made above, where the phenotypes found in nature tend to be those with largest NS sizes). Raman and Wagner studied various properties of this phenotype: They found that 98.5% of the genotypes designing this phenotype form a single connected set, meaning that any two genotypes in the set can be joined by single steps without leaving the set. They also found that this phenotype is the most robust to mutations. More generally, they found a strong positive link between robustness and evolvability in evolutionary simulations of this system, i.e. that phenotypes of larger NS sizes had mutational access to more other phenotypes, as well as being more robust to mutations. These properties have been observed in many other GP maps [182].

2.2.4.2 A model GRN linked to biochemical dynamics

Direct experimental support for degeneracy in GRN GP maps has been demonstrated: Tsong *et al* [176] found that different connectivity patterns (topologies) of gene networks ('genotypes') in yeast can give rise to the same network functional output (phenotype). A similar picture is given by Chouard [30], who reviews recent discoveries showing that many organisms have similar phenotypic traits, but radically different underlying regulatory connections.

In the same context but from a theoretical perspective, Nochomvitz and Li [136] studied GRN structure in a paper on phenotypic bias and robustness. Inspired by earlier discoveries of phenotypic bias in lattice proteins, they investigated bias in a simple model of biochemical dynamics generated by a Boolean GRN. The phenotype

here could represent the concentration-time profile of some chemical, as determined by a biochemical network. In more detail, they consider GRNs with $n = 3$ or 4 nodes, and have Boolean interactions between them defined by a matrix C, with i, j entry equal to 0, -1 or 1 representing absence of, negative, and positive regulation (respectively) of gene j on gene i. The genotypes in this model are the different interaction topologies in the network (i.e. different matrices C). The phenotypes are the different limit cycles produced, which can be of different lengths. For example, with $n = 4$ an example of a phenotype might be the 3-cycle $(0001) \to (0101) \to (1001) \to (0001)$. This model generated phenotypic bias, similar to other models we have reviewed.

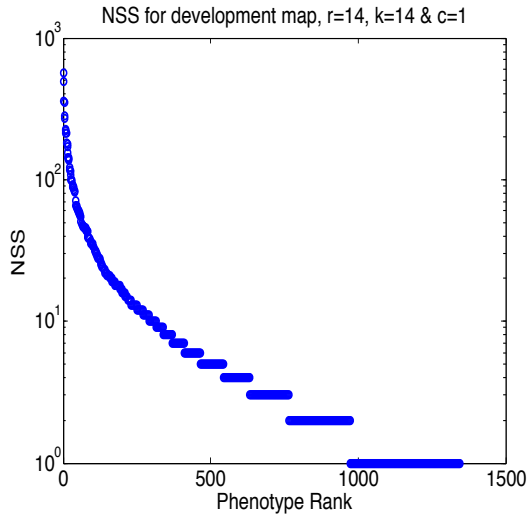
This GRN model employs discrete Boolean dynamics, but sometimes Boolean dynamics show qualitatively different behaviour when compared to the continuous dynamics defined by differential equations. In view of this, Nochomovitz and Li compared a small set of networks and their dynamics with the dynamics defined by the same topology but using differential equations, and they found that the "continuous dynamics demonstrate that the Boolean model ... exhibits the correct oscillatory phenomena". They infer that this suggests their findings are not artefacts of the Boolean model, but would also appear in more realistic biological models where continuous models are employed.

2.2.4.3 A generic GP map and developmental network

Borenstein and Krakauer [15] have published a study specifically on phenotypic bias and its implications for evolution, which is thus very relevant to this chapter. Their model is very abstract and is intended as a simple and generic GP map which consists of the typical features of GP maps (or organismic development), i.e. a genotype, a multi-component phenotype, and a mapping which incorporates gene interaction (epistasis). As an example, they suggest this framework could be seen as a model of a developmental GRN.

Their model consists of a binary vector genotype which (for example) represents a pattern of transcription factors, acting as the 'input' to the GRN, with a '1' in a

genotype vector locus denoting the presence of a transcription factor, and a '0' denoting its absence. The developmental plan network representing the interactions of these genes is given by a matrix $\mathcal{D}$, which has entries either $1, -1$, or 0 representing positive, negative or absent regulation. Phenotypes are binary vectors that represent the pattern of gene expression output by the network, and are computed by multiplying the genotype vector by the network interaction matrix, and setting each gene 'on', if its expression level is beyond a given threshold, and 'off' otherwise.



(a)

Figure 2.8: The Borenstein and Krakauer developmental GRN model. The NSS vs. rank plot for the model using a square matrix of size 14, $c = 1$. The NS sizes range smoothly and progressively from 566 down to only 1, over 1344 phenotypes. This plot has $\beta = 0.28$ and $\sigma/\mu = 2.88$.

By sampling a large number of genotypes, Borenstein and Krakauer found that these phenotypes are not uniformly represented, rather there is strong phenotypic bias, and further that many conceivable phenotypes, or "potential phenotypes", were not generated by *any* genotype. We simulated the model using the parameters which they focussed on, i.e. a size 14 square matrix and $c = 1$ (a parameter determining the density of connections in the matrix $\mathcal{D}$), see Figure 2.8. Using this size 14 matrix implies that there are $2^{14} = 16384$ genotypes.

The plot shows strong phenotypic bias with the largest NS sizes in the range of 600, with fast decay down to very small NS sizes. Quantitatively, $\beta = 0.28$ and $\sigma/\mu = 2.88$, thus the phenotypic bias is very strong.

The phenotypic bias observed was found to be robust to model alterations, such as: Altering the interaction density (c); different size and non-square matrices; choosing the matrix entries from a Normal or uniform distribution instead of only $+1, -1, 0$; and layering the process such that the phenotype of one matrix forms the genotype of another.

One of the aims of the original study [15] was to demonstrate how development (i.e. the GP map itself) plays a large role in determining biotic diversity. One qualitative feature of biology which they sought to address was why organismic morphologies occupy only a small proportion of the 'space' of possible forms. Essentially the answer suggested by their work is that phenotypic bias — naturally arising in developmental GRNs — consigns much of phenotype space to having very small NS sizes, or having *no* genotypes underlying them at all, thus strongly reducing the space of phenotypes. They note that this is different to explaining the restricted morphological variation in nature and evolutionary convergence via selection, which acts *after* the variation has already been produced. Instead these authors suggest that restricted morphological variation and convergence may be explained by properties of GP maps/GRNs, which constrains what morphological variation is actually produced. We incline to Borenstein and Krakauer's view, and the several examples of bias and reduction in potential phenotypic variation given in this review support this view also.

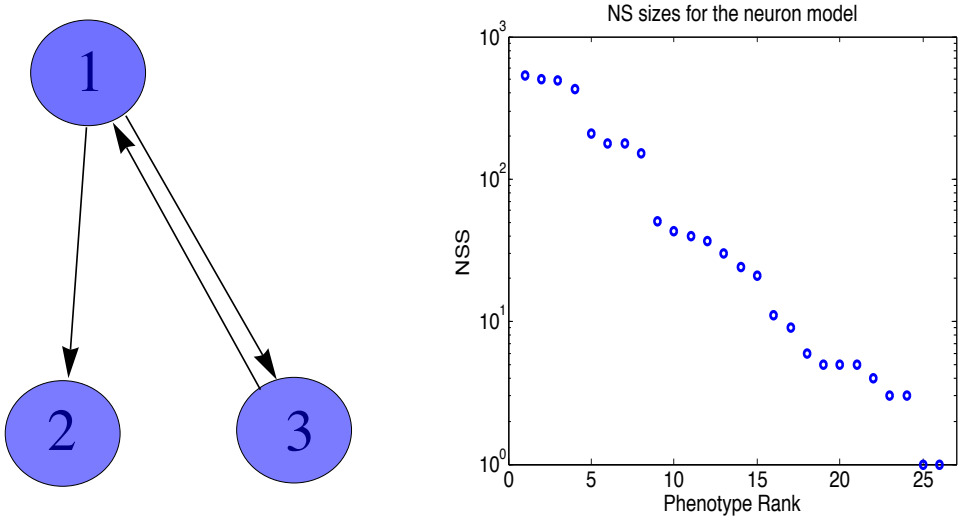
2.2.5 Development

2.2.5.1 Model of neuron development

Psujek and Beer have performed an investigation of phenotypic bias that is highly relevant to our investigation [145]. The map is a computational model of neuronal development. Due to the computational intractability of large neuron networks, the system they developed consists of only three neuron cells, and each pair of neurons i and j may have a connection arrow (neuronal connection) going from i to j, from j

to i, or none at all. The phenotype is the connectivity pattern of these neurons, and there are 64 (2^6) of these. The neurons are not equally spaced as neuron 1 is closer to neuron 2 than to neuron 3. An example phenotype is given in Figure 2.9(a).

The model is rather intricate compared to some other examples given in this review, involving nine genes (57 bases each) with more than one activation and repression section as well as coding and signalling sections. The genotypes are these genes, which (for computational purposes) only consist of A and C nucleotides. The development simulation proceeds by first spatially fixing the three cells, then the cells differentiate which results in the expression of synapse-determining surface proteins. Finally, neuronal connections are formed between cells having complementary surface proteins. We refer the reader to the original paper for the model details.



(a) (b)

Figure 2.9: Neuron development model. a) The model has only three neurons: 1, 2 and 3. Between any two neurons i and j there can be either no arrow, an arrow from i to j, and/or an arrow from j to i. The phenotypes are the connectivity patterns. An example of a phenotype is diagrammed. b) Phenotypic bias in a model of neuron development. Of the 63 non-trivial phenotypes, only 26 appeared when genotype space was sampled. The plot shows a smooth decay in NS sizes, dropping from >500 down to 1. This GP map shows bias, with $\beta = 0.4$ and $\sigma/\mu = 1.53$. Data generated by Psujek and Beer [145].

The authors observed phenotypic bias on sampling 2×10^7 genotypes, which represents a tiny proportion of the astronomical genotype space, which has size $2^{9\times57} \approx 10^{154}$. However, they state that the pattern of bias is not an artefact of insufficient sampling as the same phenotypes emerged repeatedly for different sample sizes.

The trivial phenotype having no connections between any neurons absorbed 19997025 (99.99%) of the sampled genotypes. Of the remaining 63 non-trivial phenotypes only 26 appeared at least once in the sampling; the NS size-rank plot of these is shown in Figure 2.9(b). For these data, we calculate that $\beta = 0.40$ and $\sigma/\mu = 1.53$; further only 10 (=26β) phenotypes account for 93% of genotypes (with non-trivial phenotypes). We note that a β or σ/μ value derived from partial sampling of genotype space is not likely to be representative of the true value. Hence as the sample the authors took was only a tiny fraction of all genotypes and presumably many designable phenotypes were not generated, this value cannot be compared in a meaningful way to the values obtained in the other GP map examples.

It is interesting that in real synaptic connections in worms, only a tiny fraction of conceivable network patterns are observed [5]. We speculate that perhaps such phenotypic bias as we have seen in this neuron model may contribute to explaining this observation of restricted morphological diversity.

These authors also explored local biases along mutational pathways, i.e. biases for certain phenotypes in the immediate single point mutation neighbourhood of a genotype. They observe that the local bias is often different from the global phenotypic bias, meaning that depending on the position of a genotype, different phenotypes are more/less likely to be accessible. In conclusion they suggest that phenotypic bias and path dependent local bias "could have a strong influence on the direction of evolutionary modification".

2.2.5.2 Spatial pattern formation

The final example of bias we describe is that of Khatri *et al* [97]. These authors developed a model of gene-regulation for spatial patterning in a 1-dimensional embryo.

The aim of this computational study was to elucidate the effects of phenotypic bias on evolutionary convergence, especially when population size is varied.

This genotype-phenotype model is somewhat detailed, though still a highly simplified model of pattern formation in development. The system consists of a morphogen M with concentration gradient exponentially decaying along the 1-D space, a transcription-initiator protein R (e.g. RNA polymerase), and a single transcription factor (TF) T, whose 1-D concentration profile is the phenotype (which is not discretised). This phenotype depends on how well the proteins M and R bind to (a) the DNA regulatory regions of T, and (b) each other. These binding strengths are determined by the genotype, a length 50 binary sequence. Phenotypes with proportionally higher concentrations of T in the anterior end of the 1-D space are assigned higher fitness values, and higher concentrations of T at the anterior end have a fitness cost.

The authors ran many simulations with random genotypes and for different population sizes. By reference to earlier work [161] showing that populations tend to maximise "free fitness" (mean fitness plus an entropy term), and by noting the important contribution of "mutational entropy" (i.e. phenotypic bias) to free fitness, they attribute some of the observed convergence to bias. Further they note that maxima in free fitness will not generally correspond to maxima in fitness (due to the role of bias), and hence populations may evolve towards phenotypes which are not the fittest. Regarding the effect of population size, they found that with small populations, convergence to phenotypes of larger NS sizes occurs, while with larger populations fitness effects dominate. This reflects the known fact that genetic drift has a greater influence for smaller populations, and selection a stronger influence for larger populations [45, 55]. In conclusion they state that "we find evolution can be biased toward certain phenotypes, simply as a result of being mapped onto from a larger number of genotypes, even if these phenotypes represent suboptimal solutions."

Summary of GP map Examples

Example	*Genotype*	*Phenotype*
Genetic code	Codon	Amino acid
RNA	Nucleotide sequence	Secondary structure
Protein	Amino acid sequence	Tertiary structure
Enzymes function	Amino acid sequence	Enzyme function
Protein complexes	Binary string	Polyomino shape
Developmental GRN	TF pattern	Gene expression pattern
GRN-Dynamics	Network connectivity	Limit cycles/conc. profiles
TOR signalling	Network connectivity	Conc. profiles for key proteins
Neuron model	DNA sequences	Neuronal connectivity pattern
Spatial patterning	DNA sequences	1-D morphogen conc. profile

Table 2.1: A summary table of the GP maps surveyed in this chapter. Phenotypic bias is observed in many different contexts, spanning many biological level of organisation.

2.3 Conclusion

In summary, we have reviewed many examples of phenotypic bias: A highly non-uniform distribution in the number of genotypes per phenotype in GP maps — from a wide range of biological contexts (Table 2.1). The breadth of examples implies that the bias is not simply an artefact of any particular model. In fact, it would be interesting to find and examine any examples of GP maps (of the form considered here) *without* bias, but we are yet to find one.

As a result of these examples, we *hypothesise* that highly non-uniform NS size distributions are a general feature of GP maps which:

(a) Have many more genotypes than phenotypes;

(b) Have a clear association of a set of well defined genotypes to well defined phenotypes;

(c) Have their phenotypes specified by the genotypes without too much interference from varying external factors like diet, temperature or physical activity.

To our knowledge, we are the first to make this hypothesis of ubiquitous bias in GP maps.

Some implications of phenotypic bias for evolution were reviewed, and it was found that mathematical modelling, bioinformatic and experimental studies suggest

that neutral set size (NSS) can have a significant influence on evolutionary outcomes. For example, a recent paper by Schaper and Louis [156] gives a detailed model for evolutionary dynamics with bias, showing in detail how large bias can overcome fitness advantages. Accordingly, in the cases of amino acids, RNA secondary structure, protein tertiary structures and the TOR signalling circuit, there is evidence to suggest that the phenotypes with large NS sizes are overrepresented in nature.

In consideration of the above, although currently there does not appear to be widespread awareness of bias in GP maps, and some early work of evolution theory is taken as justification for the marginal importance of bias, we suggest that bias should be considered more seriously in evolution studies. Future work will have to explore the extent of phenotypic bias by more detailed study, both from an experimental and theoretical perspective, to see how common bias is, and to further elucidate its potentially profound implications for evolutionary dynamics.

Chapter 3

The Distribution of RNA Sequences Over Secondary Structures

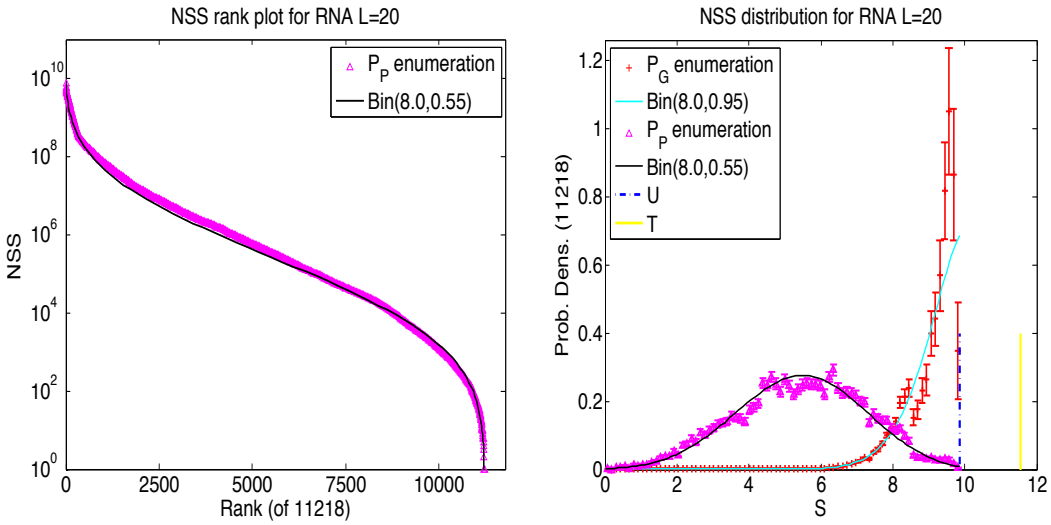
In the previous Chapter, we surveyed many examples of phenotypic bias in GP maps. The general conclusions were that bias may be a common property of GP maps, and that bias can affect evolutionary dynamics and outcomes. In this Chapter we make a more thorough investigation of one of the maps we looked at, namely the RNA sequence-secondary structure map. We suggest an analytic form of the distribution of sequences over secondary structures, and use this to quantify a range of properties of the map. Finally we compare neutral set sizes of natural RNA sequence data to our sampling and analytic predictions, and find striking agreement.

3.1 Introduction

We have surveyed many examples of bias in GP maps, but we have not investigated the precise form of any of these neutral set size (NSS) distributions. Knowing the form of the NSS distribution can allow for a detailed quantitative understanding of the nature of a given GP map, and facilitate quantitative predictions, and comparison to natural data.

In general, it is difficult to gain such a detailed quantitative understanding of a GP map, due to the requirement of being able to know how genotypes map to phenotypes. However, one system where some progress can be made is the GP mapping from an

RNA sequence to a secondary structure (SS), which we introduced in the previous chapter.



(a) (b)

Figure 3.1: NSS distribution from a full enumeration of the RNA SS found for the $4^{20} \approx 10^{12}$ sequences with $L = 20$. (a) A ranked plot of the 11218 (non-trivial) phenotypes which range in NSS from $\sim 10^{10}$ over many orders of magnitude down to 1. Additionally a rank plot corresponding to a binomial fit is given (Eq. (3.1)), which follows the data very closely. (b) The probability of finding a phenotype with a NSS of 10^S genotypes is well approximated by the binomial distribution from Eq. (3.1), with chosen values $N = 8.0$ and $p_P = 0.55$. The corresponding P_G (Eq. (3.3)) distribution matches the enumeration data less well. Data in (b) is plotted by frequency with all bin sizes equal to 0.125. The bin frequencies for the five bins of largest U are 30, 33, 32, 20 and 6. 10^U corresponds to the largest non-trivial NSS, and 10^T corresponds to the NSS of the trivial structure.

In this Chapter we build on Schaper and Louis' [156] computational folding of all possible RNA sequences for $L = 20$, using the Vienna package pioneered by Schuster and colleagues [81]. To our knowledge this is the largest system fully enumerated so far. We make the ansatz that for larger L the logarithm of the number of sequences per SS can be approximated by a binomial distribution. In other words, the NSS are distributed as 10^S, where the exponent S is distributed binomially. We show by direct sampling for longer lengths of $L = 30$ up to $L = 100$ that this relatively

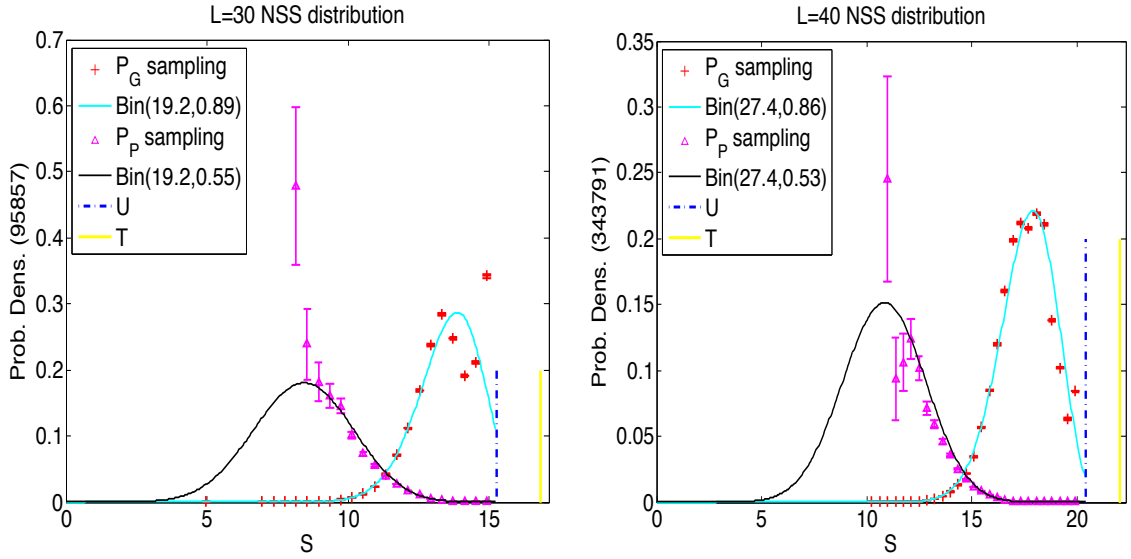
simple distribution fits the data remarkably well. We can therefore quantify the L dependence of a number of other properties of the RNA GP map, including the number of phenotypes N_P, a bias measure derived from the Shannon information of the distribution, and the size of a typical NSS.

Finally we show that the distribution of NSS for natural sequences from a non-coding functional RNA database (fRNAdb [99]) roughly follows the related log-binomial distribution that arises from randomly sampling genotypes. Even though SS is important for RNA function [21, 7, 8], and is under selection [129, 144], remarkably, the *distribution* of non-coding functional RNAs found in nature resembles that found for uniformly randomly sampling genotypes. Moreover, we find that two important functional RNA structures, the $L = 55$ type III hammerhead ribozyme and $L = 70$ tRNA aptamers have among the most typical NSS, as compared to random genotype sampling. Taken together, these results suggest that the structure of the RNA GP map, the way sequences are distributed over SS, plays a key role in the evolutionary outcomes.

3.2 Results

3.2.1 Analysing data from complete enumeration for $L = 20$ RNA

Figure 3.1 shows the results of computationally folding all sequences to SS for $L = 20$ RNA, which took about $\sim$1 year of CPU time [156]. Figure 3.1(a) depicts a log-linear plot of NSS versus rank, and illustrates that the distribution is highly non-uniform with clear phenotypic bias. The highest ranked (non-trivial) phenotypes have $\sim 10^{10}$ genotypes mapping to them, and subsequent NS sizes decay until the lowest ranked structure (rank 11218) has only 1 genotype sequence mapping to it. (Note that in the Figure we ignore the trivial phenotype with no bonds, which has 3.64×10^{11} genotypes (i.e. 33% of the total 4^{20}) mapping to it.) Figure 3.1(b) shows a complementary distribution, the probability $P_P(S)$, that a SS phenotype has as NSS size of 10^S. It shows that, when sampling over SS phenotypes, both the largest and smallest NSS



(a) (b)

Figure 3.2: Inferring the $P_P(S)$ and $P_G(S)$ distribution from extensive sampling for (a) $L = 30$ and (b) $L = 40$. The total number of samples is given in brackets on the vertical axis. The black curve (Eq. (3.1)) describes the data points well. The binomial P_G distribution (Eq. (3.3)) diverges from the binomial form close to U, and hence the binomial fit diverges from the data for structures of largest NSS. The vertical lines indicate the largest value (U) of S for non-trivial structures and the trivial structure NSS. Error bars are calculated by taking the square root of the number of sampled SS in a bin. For many bins the error bars are smaller than the symbols.

are fairly rare, and that the most likely NSS per SS phenotype is close to half the maximum NSS.

The amount of work needed for a full enumeration grows rapidly with increasing L. Not only are there exponentially more sequences to fold, but longer sequences take more computational resources to find their minimum free energy SS. This hinders direct approaches to finding the NSS distribution for larger L. Some progress has been made using only two-letter alphabets of GC or AU for a short length ($L = 30$) [74] but extrapolation to full four letter alphabets is difficult. Another investigation used a more coarse grained representation of RNA [160], finding a generalised Zipf's law for the distribution, but that was was not supported by the later work of Ref. [74]. Thus, making progress on finding $P_P(S)$ for larger L via exhaustive enumeration will be difficult for lengths far beyond what we have done here. Instead, sampling methods [91] and approximations are called for.

3.2.2 Approximating NSS distributions of SS phenotypes with a log-binomial form

Motivated by the form of the plots for $L = 20$, we make the following central *ansatz*: The full distribution of $S = \log_{10}(\text{NSS})$ can be approximated as

$$P_P(\log_{10}(\text{NSS}) = S)) = \text{Bin}(\frac{SN}{U}, N, p_P) \tag{3.1}$$

where

$$\text{Bin}(r, N, p) = \binom{N}{r} p^r (1-p)^{N-r} = \frac{N!}{(N-r)!r!} p^r (1-p)^{N-r} \tag{3.2}$$

is the binomial distribution, 10^U is the largest non-trivial NSS, and N and p_P are a parameters that are fit to the distribution (see Methods). We do not consider the trivial structure because for small L its NSS is anomalously large in relation to the rest of the structures. For $L = 20$, Eq. (3.1) with chosen values $N = 8.0$ and $p_P = 0.55$ describes the exact distribution from full enumeration rather well, as can be seen in Figure 3.1(a) and (b). Note that although strictly speaking the binomial distribution of Eq. (3.2) is only defined for integer values of N and r, it can be evaluated for non-integer values in a standard way.

If instead of sampling over SS phenotypes we sample uniformly over random genotypes (which we will call *genotype sampling*), then we are much more likely to generate SS with large NSS. More precisely, the distribution of the NSS of the SS so found will scale as $P_G(S) \propto P_P(S)10^S$ and (by taking normalisation into account) can be rewritten in another binomial form as:

$$P_G(\log_{10}(\text{NSS}) = S) = \text{Bin}(\frac{SN}{U}, N, p_G) \tag{3.3}$$

where p_P and p_G are related by

$$p_G = \frac{p_P 10^{U/N}}{1 - p_P + p_P 10^{U/N}} \tag{3.4}$$

and

$$p_P = \frac{p_G 10^{-U/N}}{1 - p_G + p_G 10^{-U/N}} \tag{3.5}$$

These two distributions, $P_P(S)$ and $P_G(S)$, are hence closely related; fixing one fixes the other.

Returning to the distribution for $L = 20$ shown in Figure 3.1, while P_P appears to follow the data very closely, the corresponding P_G distribution does follow the data somewhat, but less well. In particular, Eq. (3.4) implies P_G should follow Bin(8.0,0.95) which continues to ascend as S approaches U; in contrast the data reaches a peak and decays close to U.

3.2.3 Comparing the binomial forms to sampling

For longer $L > 20$ we must use sampling techniques (described in Methods, Section 3.4). The genotype sampled distribution $P_G(S)$ is much easier to determine than the phenotype sampled one $P_P(S)$ because it is straightforward to randomly choose genotypes, but hard to choose phenotypes in a uniformly random way.

For L not much larger than 20, as the space of genotypes and phenotypes is still small enough that we can get reasonable approximations to the P_P distribution via extensive sampling. Figure 3.2 shows results for (a) $L = 30$ and (b) $L = 40$. As can be seen, by choosing appropriate values of N and p_P for the distribution P_P, good

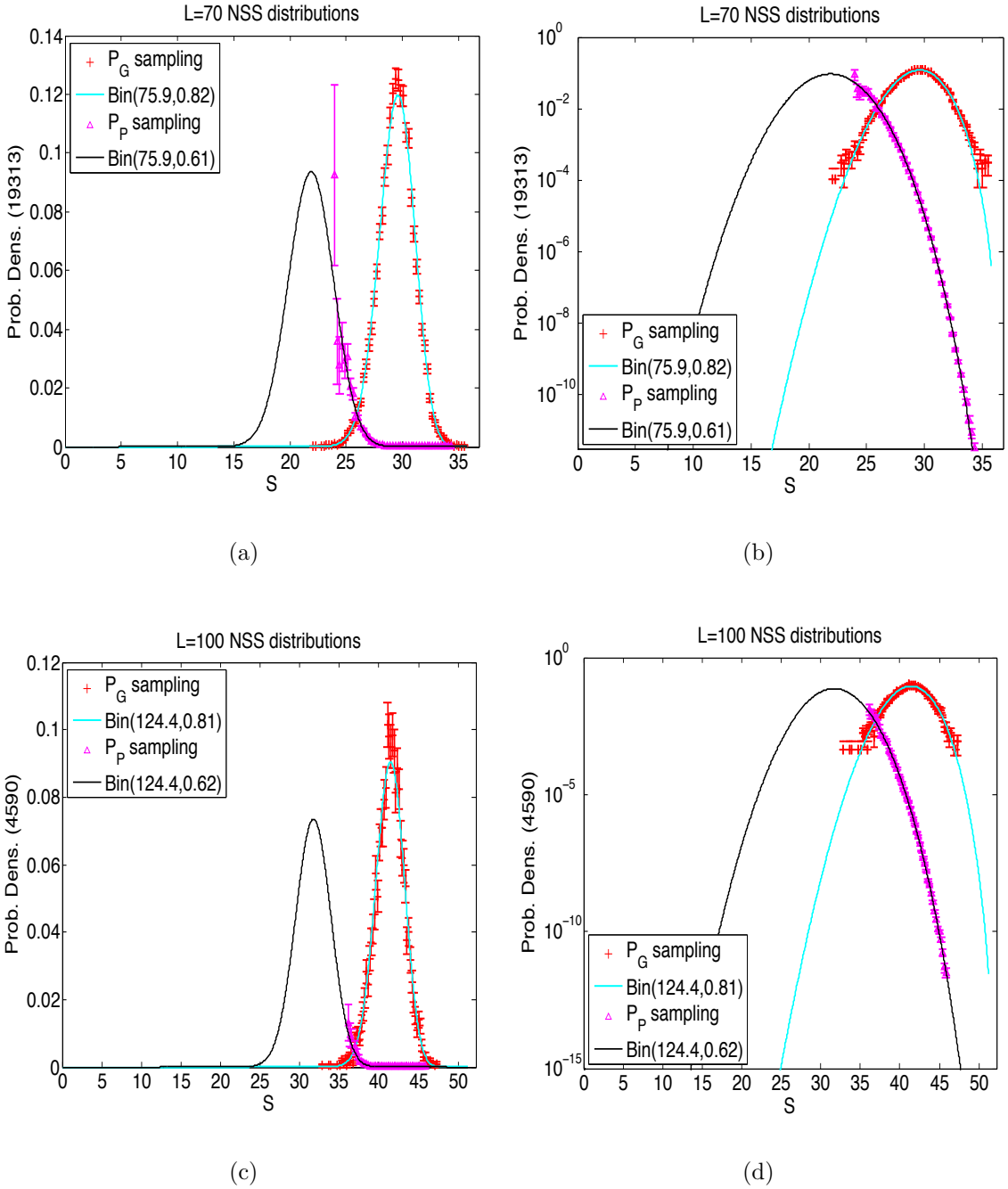


Figure 3.3: Comparison of the log-binomial forms of Eqs. (3.1) and (3.3) to directly sampled distributions of RNA neutral set sizes (NSS). (a) Linear axis plot and (b) log axis plot for $L = 70$; (c) Linear axis plot and (d) log axis plot for $L = 100$. The sampled distribution $P_G(S)$ comes from choosing random genotypes to find a SS and then calculating the NSS (see Methods, Section 3.4). The number of samples used to create the distribution $P_P(S)$ is given in brackets on the y-axis.

fits to the sampled data can be made. As expected from sampling, only SS with large NSS appeared.

As with the $L = 20$ data, we find that the P_G distribution fits the data less well for NSSs close to U. Indeed, for small $L \lesssim 40$ we find the P_G distributions appear less smooth, with the distribution of largest NSS having more structure. For larger L, the distributions become more smooth, and our predictions hold well. So, for example, in Figure 3.3 we depict sampled data for $L = 70$ and $L = 100$ on a linear and log scale, and show that these closely match our analytic forms for $P_P(S)$ (Eq. (3.1)) and $P_G(S)$ (Eq. (3.3)) respectively over several orders of magnitude. The measurement of U and fitting to N, which fixes both analytic distributions for a given L, is described in Methods (Section 3.4).

Despite these apparent successes, it is clear that finding phenotypes with smaller NSS via sampling is hard, so that at present we have a good sampling of $P_G(S)$, but only can sample the large S portion of $P_P(S)$. However, given how well our simple ansatz works for predicting $P_G(S)$, and given that for $L = 20$ RNA the binomial form $P_P(S)$ works well for the *full* range of structures, we expect the full $P_P(S)$ to be at least similar if not very close to the one we show in Figure 3.3.

3.2.4 Scaling of distribution properties with RNA length L

By sampling over the genotypes for different lengths L we are able to extract a number of properties of both distributions as a function of L. As detailed in the Methods section, we find the maximum $\log_{10}(\text{NSS}) = U$ to scale with L as

$$U = (0.514 \pm 0.009)L - (0.2 \pm 0.5) \tag{3.6}$$

Similarly for $T = \log_{10}(\text{NSS})$ of the trivial structure, we find

$$T = (0.5166 \pm 0.0009)L + (1.33 \pm 0.06) \tag{3.7}$$

So U is very close to T for large L, and the trivial structure has larger NSS than any non-trivial SS ($T > U$). The probability that a randomly chosen sequence adopts the

trivial structure, i.e. that it does not fold stably to a SS follows from Eq. (3.7):

$$P_T = 10^T/4^L \approx 21.4 \times 0.82^L \tag{3.8}$$

which vanishes exponentially to zero for increasing L. This also shows that for larger L, whether or not the trivial structure is ignored in the binomial fit makes little difference.

Another important property of the distribution that we measure is the mean value of $S = \log_{10}(\text{NSS})$. By sampling over random genotypes we find that:

$$\bar{S}_G = (0.399 \pm 0.0014)L + (1.48 \pm 0.09) \tag{3.9}$$

where the notation $\bar{X}_G$ means the mean of a variable X sampled over genotypes. Similarly $\bar{X}_P$ is the mean when sampling over phenotypes.

From Eq. (3.1) and Methods we find that for large L, $\bar{S}_P = p_P U \approx 0.63 \cdot 0.514L = 0.32L$ and $\bar{S}_G \approx 0.399L$, so that $\bar{S}_P/\bar{S}_G \approx 0.80$ becomes independent of L. The variance of a binomial distribution is given by $\sigma^2 = Np(1-p)$. Thus, assuming p_P and p_G have reached their asymptotic values of 0.63 and 0.78 respectively (Methods), the two (rescaled) standard deviations, $\sigma_P = \sqrt{Np_P(1-p_P)}U/N$ and $\sigma_G = \sqrt{Np_G(1-p_G)}U/N$ scale as:

$$\sigma_P \approx 0.19\sqrt{L} \tag{3.10}$$

$$\sigma_G \approx 0.16\sqrt{L} \tag{3.11}$$

respectively. Thus for larger L we see analytically what could be observed qualitatively in Figure 3.3: The $P_G(S)$ distribution is slightly narrower than the $P_P(S)$ distribution. For large L, $\sigma_P/U \approx 0.37/\sqrt{L}$ and $\sigma_G/U \approx 0.31/\sqrt{L}$. In other words, as L increases, relative to total range, $[0, U]$, both distributions become more sharply peaked, and the $P_G(S)$ distribution highlights SS phenotypes that are further away from the mean of the $P_P(S)$ distribution. For example, with $L = 100$, $\bar{S}_G = 41.4$ and by Eq. (3.10), $\sigma_G \approx 1.6$. In other words, for $L = 100$, 95% of SS structures found by genotype sampling will have $S = \log(\text{NSS})$ in the range [38.2,44.6] (i.e. 41.4 ± 3.2) which is very narrow compared to the range of $[0, 51]$ for the full distribution. This

is an important find: Despite the astronomical number of genotypes and phenotypes in the evolutionary space for $L = 100$, the GP map structure biases the evolutionary search in genotype space to phenotypes in a small region of phenotype space.

A summary of the scaling forms of various distribution properties in the large L limit are also depicted in Table 3.1.

3.2.5 Why not use the log-normal instead of log-binomial?

It is well known that the binomial distribution Eq. (3.2) is closely approximated by the normal distribution for large N and fixed p [151]. Since our fits have $N \gtrsim 20$ so that the two distributions are expected to be very similar, the reader may wonder why we use the somewhat esoteric log-binomial distribution in place of the familiar log-normal distribution. The answer is that $P_G(S)$ and $P_P(S)$ are related through exponential weighting of the tails of the distribution and so tiny differences in the distributions are magnified. When both distributions are transformed by an exponential weighting $P_G(S) \propto P_P(S)10^S$, the forms of the new distributions are notably different. This transformation affects the binomial by increasing the mean from $N \cdot p_P$ to $N \cdot p_G$, and the variance decreases from $N \cdot p_P \cdot (1 - p_P)$ to $N \cdot p_G \cdot (1 - p_G)$. Hence the shape of the distribution changes. As for the normal distribution $\mathcal{N}(\mu, \sigma^2)$ and applying the weighting $e^{\alpha x}$, then it is easy to show that this weighting affects the distribution by changing it to $\mathcal{N}(\mu + \alpha\sigma^2, \sigma^2)$ i.e. a normal distribution with greater mean, but the same variance. As such, in the normal case, the shape of the distribution remains unchanged, but is simply shifted. Hence, since our sampling has shown that the distribution shapes do change, and in a manner expected by the binomial form, then the normal fit is inappropriate.

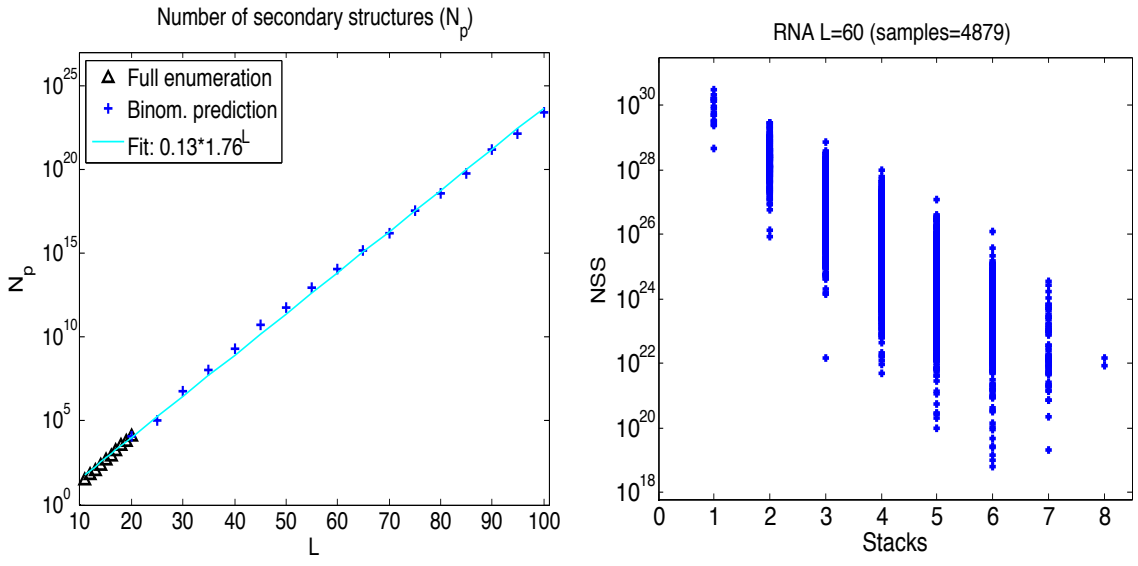
3.2.6 Estimating the number of secondary structures

Another important property of the RNA GP map is the total number of designable secondary structures N_P for a given length L, that is, the number of structures having at least one associated sequence. It is possible to use combinatorics to count all possible ways that sequences can be bonded, which leads to the approximate large

Scaling of key quantities for RNA neutral set size distributions

Quantity	*Large L scaling form*
Total number of genotypes	$N_G = 4^L$
Total number of SS phenotypes	$N_P \approx 0.13 \times 1.76^L$
Mean NSS	$4^L/N_P \approx 7.7 \times 2.27^L$
Largest NSS	$10^U \approx 0.7 \times 3.27^L$
NSS near peak for phenotype sampling	$10^{\bar{S}_P} \sim 2.1^L$
NSS near peak for genotype sampling	$10^{\bar{S}_G} \sim 2.5^L$
Shannon entropy of distribution (in bits)	$H \approx 0.675L - 4.92$
Bias parameter	$\beta = \frac{2^H}{N_P} \approx 0.26 \times 0.91^L$
Number of 'relevant' SS phenotypes	$\beta N_P \approx 0.033 \times 1.596^L$
$\log_{10}$(NSS) for the trivial structure	$T = 21.4 \times 3.29^L$
Probability to sample trivial structure	$P_T = 21.4 \times 0.82^L$

Table 3.1: The large L scaling of some key quantities that characterise the distribution of neutral set sizes (NSS) for RNA secondary structures (SS).



(a) (b)

Figure 3.4: (a) The number of SS, N_P, can be estimated using the binomial fit to be $N_P \sim 1.76^L$. (b) Log NSS correlates quite strongly (linear core. coeff. $R = 0.82$) with number of stacks (stems) for $L = 60$ RNA. A similar correlation is found for other lengths.

L scaling form $N_P \approx 1.104 L^{-1.5}(2.618)^L$ [168, 172]. However, this assumes that all nucleotide pairings are allowed. A different combinatorial approximation that includes physically motivated constraints like a minimal loop size of three and no isolated pairs yields $N_P \approx 1.4848 L^{-1.5}(1.8488)^L$ [74]. A drawback of direct combinatorial analysis is that while it can be used to study the space of *conceivable* structures, it is not immediately clear how this relates to actually designable structures, whereas this latter set is of greater biologically relevance.

We studied the data for complete enumerations with $L = 11, 12, \ldots, 20$ [156], and used the binomial approximation with parameters fit as described above to estimate N_P for longer L (assuming the Vienna package and standard settings). This was done by observing that the mean NSS is $4^L/N_P$, and so

$$(1 - P_T)4^L = \sum_{s=0}^{U} N_P P_P(s) 10^s \tag{3.12}$$

from which we can derive the expression

$$N_P = \frac{(1 - P_T)4^L}{(1 - p_P + p_P 10^{U/N})^N} \tag{3.13}$$

To evaluate N_P with this equation, we can use the analytic forms for P_T, U and N, in addition to a numerical estimate for p_P using the relatively simple expression for p_G in Methods (Eq. (3.27)) combined with Eq. (3.5).

In addition to an exact expression, it would be useful to have a simple analytic expression for N_P; however this is not straightforward, in part because we do not have a simple analytic form for p_P. Nonetheless we can do two things: Firstly, we can make a large L scaling estimate by using asymptotic values for the quantities P_T, U, N and p_P. So, for large L we have $P_T \approx 0$, $p_P \approx 0.63$ (Methods) and $10^{U/N} \approx 2.03$, hence

$$N_P \sim 1.73^L \tag{3.14}$$

Having derived this scaling, we caution that it may only apply for very large L, where our approximations and fits no longer hold accurately.

Secondly, by fitting, we can get a simple analytic estimates of N_P for L in of the range of lengths we focus on here: In Figure 3.4(b) we plot values of N_P against L and make an exponential fit

$$N_P = (0.13 \pm 0.04) \times (1.760 \pm 0.007)^L \tag{3.15}$$

While we used the full-enumeration value of N_P for $L = 20$ to make this fit, it is reassuring that the binomial predicted value (also plotted) is very close to the true value obtained from full enumeration, and that Eq. (3.15) fits the data for the whole range of L. Additionally, this fit is broadly in agreement with the purely combinatorial approaches cited above.

In sum, we have a precise but analytically cumbersome expression for N_P (Eq. (3.13)), as well as a close analytically simple expressions for N_P (Eq. (3.15)). Due to the simple form and close quantitative accuracy, we will use the exponential fit (Eq. (3.15)) in this work when analytic scalings are required.

3.2.7 Correlation between number of stacks and $\log_{10}$(NSS)

In Figure 3.4(b) we plot the number of stacks (i.e. contiguous base pairs) in sampled SS, which shows a strong correlation with $\log_{10}$(NSS). Such correlations are interesting because they imply, for example, that for a fixed L, sampling over phenotypes will generate a different average number of stems than sampling over genotypes would. In particular, this correlation implies that by sampling genotypes, a bias for few stacks will be observed, as compared to the distribution over the full range of phenotypes.

By way of motivation for future work, it is interesting that it appears this correlation may be used to make predictions about the distributions of NSS. Specifically, if we assume this correlation holds for the full range of stacks and NSS, then we expect to be able to predict p_P as follows: A linear fit to the data yields $S = 30.3 - 1.17K$, where K denotes the number of stacks. Hence, it is reasonable to presume that $\bar{S}_P = 30.3 - 1.17\bar{K}_P$. From the combinatorial analysis of Hofacker *et al* [82], we have the estimate $\bar{K}_P \approx 0.17L = 10.3$. Using $L = 60$ gives $\bar{K}_P \approx 10.3$ and (using the correlation) $\bar{S}_P = 18.25$. These values imply an estimate of $p_P = 18.25/U = 0.60$.

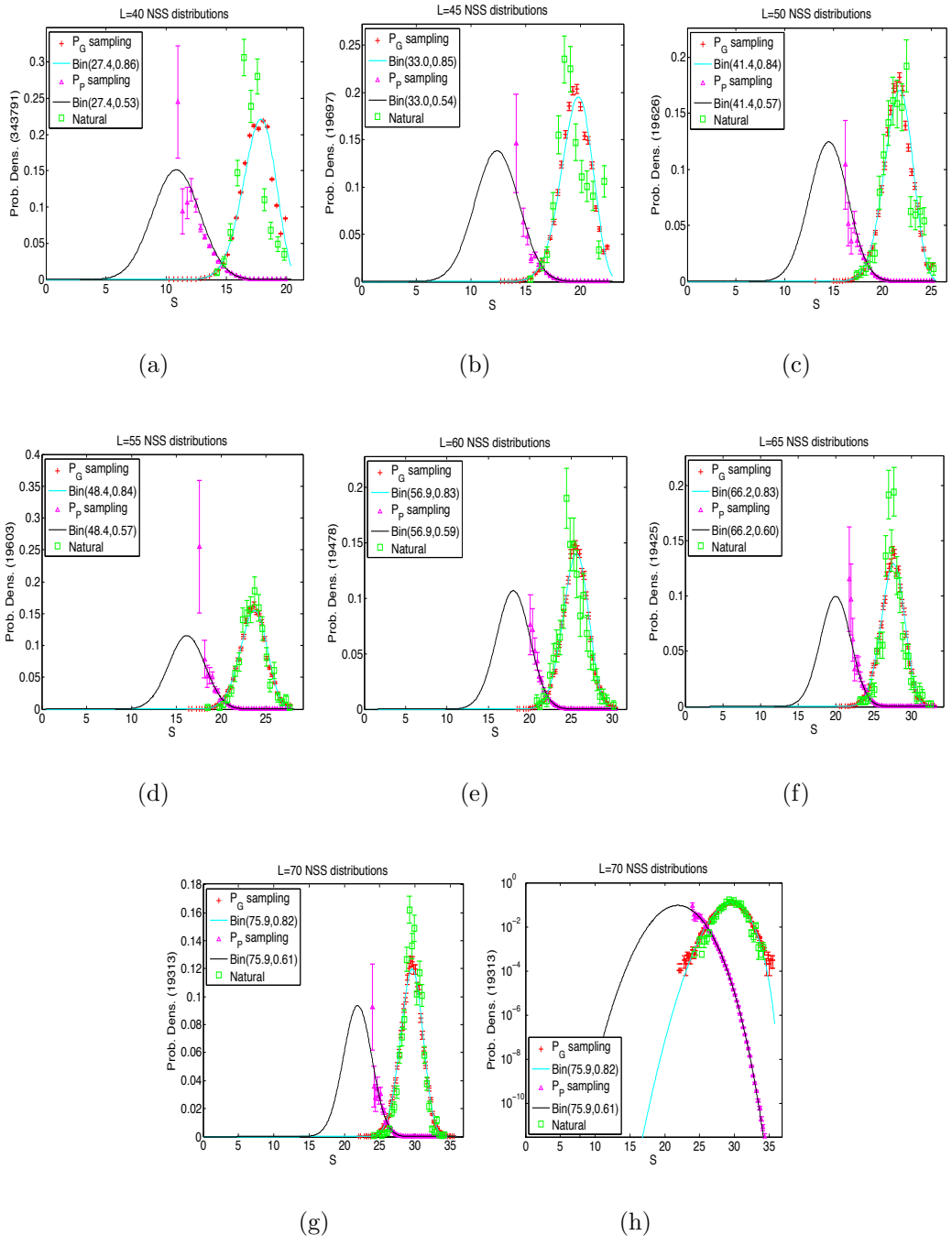


Figure 3.5: The NSS distributions of natural functional RNA are remarkably close to the NSS distributions of randomly genotype sampled RNA and the binomial form of Eq.(3.3). Natural sequences were taken from the fRNAdb database [99], and computationally folded. (a) $L = 40$ (658 structures); (b) $L = 45$ (477 structures); (c) $L = 50$ (472 structures); (d) $L = 55$ (501 structures); (e) $L = 60$ (350 structures); (f) $L = 65$ (535 structures); (g) $L = 70$ (2263 structures); and (h) The same $L = 70$ data as (g), but on a log-scale. The random sampling of genotypes that were folded consists of ∼20000 or more sequences, indicated in brackets on the y-axis label.

This estimate is very close to the binomial estimate we have for $L = 60$ of $p_P = 0.59$. Performing the same analysis for $L = 30$ (where the fewer possible stacks makes the correlation harder to infer), we find an estimate of $p_P = 0.58$, whereas the value used in the binomial was 0.55. It is tempting to speculate that this method will be more generally successful, and that possibly some of the binomial parameters may be approximated directly from combinatorial analysis.

3.2.8 Why binomial?

We have shown that the binomial form provides a good fit to the distributions $P_G(S)$ and $P_P(S)$. A natural question to ask is why this form should work. While we do not have a full answer to this question, we nonetheless suggest a perspective which motivates a binomial, or at least a unimodal tightly peaked distribution, for $P_P(S)$ (and hence $P_G(S)$).

In the previous subsection we showed a fairly strong linear correlation between S and the number of stacks K in an RNA structure. Hence, to first approximation, if we can show that the full $P_P(K)$ distribution of K is roughly unimodal and tightly peaked (with a roughly binomial/normal form), then this would motivate a similar distribution for $P_P(S)$.

To analyse the $P_P(K)$ distribution of stacks, we use Hofacker *et al* [82] exact analytic recursion formula for the number of RNA structures of length L with K stacks. We implemented their formula[1] for various lengths L, see Figure 3.6. As can be seen in this Figure, a unimodal and narrowly peaked distribution which is roughly mid-way between the minimum value of K (which is 0) and maximum value of K (which is $\sim L/3$) can be seen for all lengths.

It is worth pointing out that these analytic forms derive from counting stacks in all conceivable[2] RNA structures, rather than all possible structures with at least one

[1]We had to make two simple minor corrections to the formula, both of which were confirmed by Peter Stadler, who was one of the authors of the paper Hofacker *et al*'s [82]. The corrections were to set $Z_n(0) = 0$, and to set $Z_n(b) = 0$ if $n < m + 2$ (see original paper for the definitions of $Z_n(b)$ and m).

[2]For the analytics we set the minimum loop size to be 3, as this was the size we set for our computational folding.

sequence mapping to them, as we have studied computationally in this chapter. Having said that, given the fact that the distributions plotted are so strongly peaked, it would be surprising if the collection of structures generated by computational folding had very different properties to these conceivable ones. Further, we are only interested in a qualitative rough distribution for $P_P(K)$, which would give an indication of the distribution for $P_P(S)$.

In sum, from this brief analysis, it appears a binomial-type $P_P(K)$ distribution for the number of stacks is probable, and hence roughly binomial/normal distributions for $P_P(S)$ and $P_G(S)$ have been motivated.

3.2.9 Quantifying phenotypic bias

A motivating broader goal for this work is to obtain a quantitative appreciation of phenotypic bias in GP maps. One aspect to study is a quantitative measure of the strength of phenotypic bias. To proceed, we first recall the bias or β ratio which we introduced in the last Chapter (Box 3): The Shannon entropy H (in bits) of the distribution of phenotype probabilities is

$$H = -\sum_{k=1}^{N_P} P(p_k) \log_2 \left(P(p_k) \right) \tag{3.16}$$

where $p_1, ..., p_{N_P}$ are the N_P different phenotypes, and $P(p_k)$ is the probability of choosing p_k via a random genotype sequence. Notice that $P(p_k)$ is directly proportional to the NSS of phenotype, i.e. $P(p_k) = \mathrm{NSS}(p_k)/4^L$, so that it immediately follows that

$$H = 2L - \log_2(10)\bar{S}_G \tag{3.17}$$

and hence by Eq. (3.9) the entropy grows linearly as

$$H \approx 0.675L - 4.92 \tag{3.18}$$

Alternatively, we can write

$$2^H = \frac{4^L}{10^{\bar{S}_G}} \approx 0.0331 \times 1.596^L \tag{3.19}$$

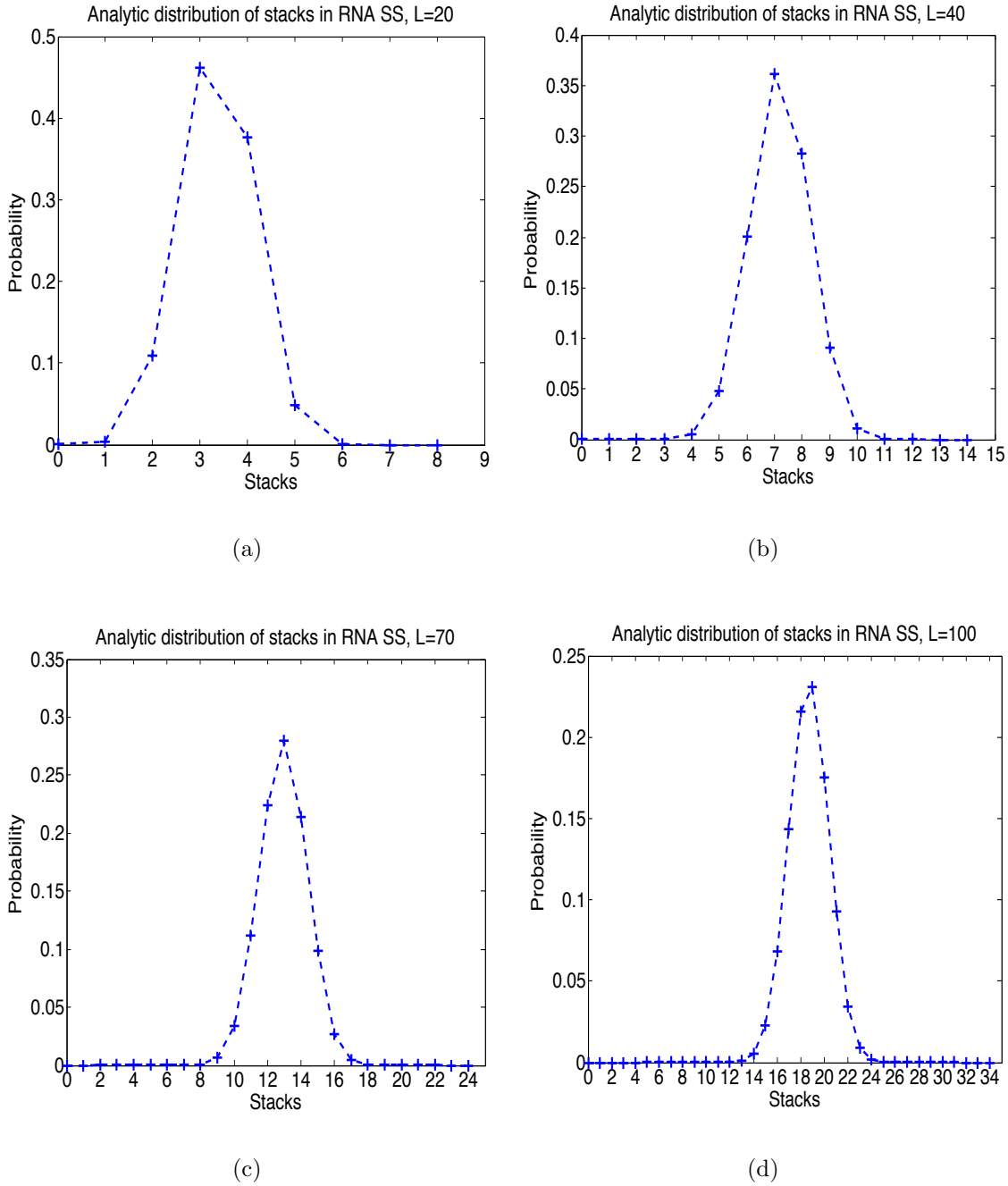


Figure 3.6: Analytically derived probability distributions (P-sampling) of stacks for RNA secondary structures, with (a) $L = 20$; (b) $L = 40$; (c) $L = 70$; and (d) $L = 100$. Notice that the distributions are uni-modal and progressively more narrowly peaked, with increasing L, and qualitatively of a binomial/normal form.

The exponential of the entropy is used in statistical physics and information theory [128] as a measure of the effective or typical number of states, or in this case the typical number of phenotypes. For example, in the case of all equal NS sizes, we have $2^H{=}2^{\log_2(N_P)}{=}N_P$, reflecting the fact that all phenotypes are well represented in terms of NSS; and for the case where one phenotype accounts for nearly all the genotypes we have $2^H{\approx}2^0{=}1$. We therefore define (as in Chapter 2) the *bias ratio* $\beta \in (0,1]$

$$\beta = \frac{2^H}{N_P} \tag{3.20}$$

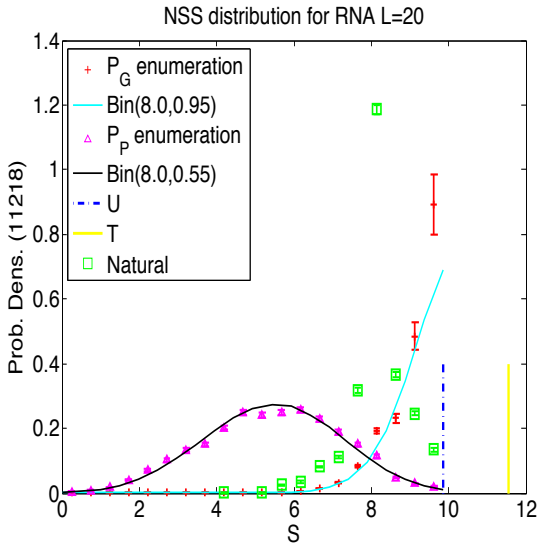
which can be interpreted as the ratio of the effective or typical number of phenotypes to the total number of phenotypes. If $\beta{\approx}1$ then the genotypes are fairly equally distributed between phenotypes, and if $\beta{\approx}0$ then the majority of genotypes map to a tiny fraction of phenotypes (i.e. the bias is very strong).

To make this measure concrete, consider the same $L = 20$ RNA as in Box 3, where the majority (88%) of (stably folding) genotypes map to a fraction of $\beta{=}0.06$ of structures (ignoring the trivial structure). As $0.06 \ll 1$, this gives a quantitative indication of the non-uniformity in NS sizes seen in Figure 3.1(a). Other examples of β ratio values for examples maps were given in the previous Chapter.

Phenotypes with above average NSS (i.e. NSS $> 4^L/N_P \equiv \overline{\text{NSS}}$) are known as 'frequent' phenotypes [182, 74]. From computational work on RNA SS, it has been observed that as the length L increases, the proportion of phenotypes which are frequent structures *decreases*, while the proportion of genotypes that map to these frequent structures *increases* [74]. Consequently it has been suggested [182, 74] that as L grows very large, nearly all the genotypes will map to a vanishing fraction of phenotypes (i.e. the 'frequent' ones). We can quantify these earlier observations. The fraction of N_P represented by these typical phenotypes roughly equals:

$$\beta \approx \frac{0.0331 \times 1.596^L}{0.13 \times 1.76^L} \approx 0.26 \times 0.91^L \tag{3.21}$$

Indeed, for large L, only a tiny fraction ($\beta \ll 1$) of possible RNA SS absorb the majority of all genotypes. For example, for $L = 50$, $\beta \approx 0.002$ and for $L = 100$, $\beta \approx 2 \times 10^{-5}$, a tiny fraction. Using the binomial fit, we have made an estimate of the



(a)

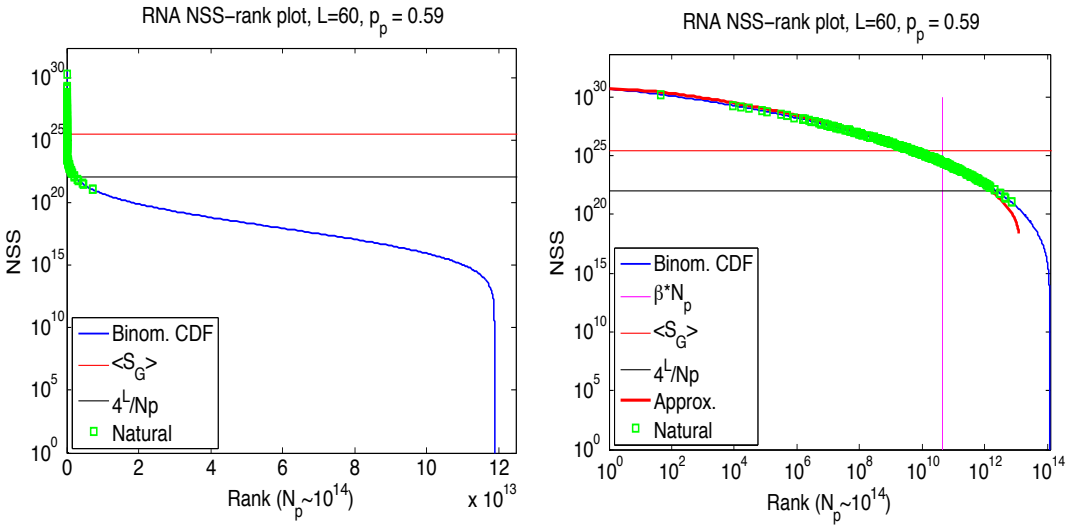
Figure 3.7: Natural $L = 20$ data (14350 structures) of RNA. The natural data is clearly biased to larger S values, and follows the distribution obtained from complete enumeration somewhat well, but less closely than for $L > 40$.

fraction of sequences folding to a fraction β with $L = 60$: We estimate $N_p = 1.2 \times 10^{14}$, and $2^H = 5 \times 10^{10}$, so $\beta = 4 \times 10^{-4}$, while roughly 75% of all genotypes map to this set of typical SS. Similar percentages of sequences mapping to the top βN_P structures were found for other lengths.

It is worth highlighting that the total number of possible SS grows exponentially with L as well: The number of typical phenotypes also grows as $\beta N_P = 2^H \approx 0.03 \times 1.596^L$ (and notice that this expression does not rely on our estimates for N_P). For $L = 50$ we find $\beta N_P \approx 5 \times 10^8$ phenotypes, while for $L = 100$, $\beta N_P \approx 7 \times 10^{18}$. Thus the actual numbers of SS phenotypes, 'typical' or not, remains incredibly large.

3.2.10 Natural functional RNA secondary structures have very similar NSS distribution to random RNA

We have so far have developed an analytic approximation to the distribution of NSS, and shown that it fits sampled data very well. An intriguing next question is to ask



(a) (b)

Figure 3.8: Rank plots for $L = 60$ RNA generated by the binomial fit, the CDF of the analytic binomial form of $P_P(S)$, cf. Eq. (3.22). (a) Full graph of all SS. (b) The corresponding log-log plot. Although there are $N_P = 1.2 \times 10^{14}$ possible phenotypes, $\beta = 0.0004$ and $\beta N_P = 5 \times 10^{10}$ take up the majority (in this case $\sim 75\%$) of the sequences (genotypes). Also shown (thick red line) is an approximation to the binomial CDF, cf. Eq. (3.24). Note that $< S_G >$ in the legend denotes $\bar{S}_G$. Roughly 31% of the natural structures have a smaller NSS than the smallest NSS in the 'typical' set.

how this relates to natural RNA SS.

To compare to natural RNA we took all available sequences from a non-coding functional RNA database (fRNAdb [99]) for lengths $L = 40, 45, 50, 55, 60, 65$ and 70. For each SS, we calculated the NSS and in this way built up a distribution (see Methods). As can be seen in Figure 3.5, there is a remarkable similarity between the natural, sampled, and analytic distributions. So not only are the structures of large NSS overrepresented in nature, as suggested previously in the literature, but even more strikingly, the full NSS *distributions* are very close to those obtained by randomly sampling over genotypes. It is worth stressing also that the natural structures are not those simply of largest NSS, but those most typical of random genotypes. In the Appendix to this chapter, we examine in some detail the collection of RNA that we have analysed from the fRNAdb.

For completeness, we also extracted all data for RNA of $L = 20$ from the same database. The distribution of NSS is plotted in Figure 3.7 (but cf. Figure 3.1), and shows that while the natural data and random samples are somewhat similar, the closeness is not so striking as for longer RNA of $L > 40$ (above). While a molecule like tRNA has a secondary structure that is important for function, it is not clear that these very small RNA have meaningful secondary structures that influence their role in an organism, and hence SS may be an artificial phenotype of these sequences (see the Appendix to this chapter for more discussion). Finally, it may also be true that selection can more easily find less frequent structures at smaller L (when the evolutionary search space is smaller) than for larger L.

3.2.11 NSS vs. rank plots

Figure 3.8(a), which is complementary to Figure 3.5(d), depicts the NSS vs. rank for the full distribution and the natural data for $L = 60$ (cf. also Figure 3.1(a)). The rank plot is generated by observing that a structure of NSS $S = s$ would have rank

$$\mathrm{Rank}(s) \approx 1 + N_P \sum_{r=k}^{N} \binom{N}{r} p_P^r (1 - p_P)^{N-r} \tag{3.22}$$

where $k = \lfloor sN/U + 0.5 \rfloor$, i.e. rounding to the nearest integer. The "1" term in Eq. (3.22) ensures that for $s = U$, the assigned rank is 1 rather than 0. Further, this rank function gives only an approximation to the true rank, as in this discretised binomial form there are only $N + 1$ different values r can take, so the rank will only be given roughly. Hence, if a NSS vs. rank plot is made using Eq. (3.22), then it will have 'steps' or plateaus in it. Knowing an approximate rank for a structure may well be sufficient, as the precise value of the rank may not be important. Having said that, if a more refined estimate of the rank is desirable, then the rank may also be estimated by summing over a more fine grained partitioning of the NSS axis, and using non-integer values of r. That is,

$$\text{Rank}(s) \approx 1 + N_P \frac{\sum_{t=k}^{N} \binom{N}{t} p_P^t (1-p_P)^{N-t}}{\sum_{t=0}^{N} \binom{N}{t} p_P^t (1-p_P)^{N-t}} \tag{3.23}$$

with $t = 0, \delta t, 2\delta t, 3\delta t, ..., N$ and where δt may be chosen to be arbitrarily small. Hence, to find k we round sN/U to the nearest integer multiple of δt. Notice that in Eq. (3.23) we have included a normalising factor, which is now needed as the summation over $t = 0, \delta t, 2\delta t, 3\delta t, ..., N$ will not necessarily equal 1.

Additionally in Figure 3.8(a) we have added an approximation to the rank function vs. NSS, shown in red, which is very close to the binomial CDF for the large NSS region, i.e. the region of typical structures. The approximation follows from noting that for $k \gg N \cdot p_P$, the summation in the numerator of Eq. (3.23) is dominated by the largest term, and hence the rank plot for large NSS structures is approximated by

$$\text{Rank}(s) \approx 1 + N_P \frac{\binom{N}{k} p_P^k (1-p_P)^{N-k}}{\sum_{t=0}^{N} \binom{N}{t} p_P^t (1-p_P)^{N-t}} \tag{3.24}$$

where k is rounded as before.

Returning to the natural data, for $L = 60$, the natural RNA only occupy the top ranked SS. Furthermore, those SS that take up the bulk of the sequences are only a very small fraction β of the total, as shown in Figure 3.8(b), where a log-log plot helps highlight the scaling of the natural RNA with rank. Also shown are $\bar{S}_G$, the mean number of sequences per structure, $4^L/N_P$, and the number of typical structures,

βN_P. The combined NSS size of these βN_P is $\sim 75\%$ of all genotypes, showing that this measure is a good predictor of the relevant states explored.

3.2.12 The secondary structures of the type III hammerhead ribozyme and tRNA have among the most typical NSS

The hammerhead ribozyme is a small catalytic RNA motif capable of (self-) cleavage. It has been extensively studied in vitro and in vivo, and appears so often in all three kingdoms of life and in viruses that it has been recently termed ubiquitous [79]. We examined the 11 type III hammerhead ribozymes of $L = 55$ which appeared in the natural data set, testing whether the NS sizes of the structures were especially large or small. Interestingly, we found that these hammerhead ribozyme SS are near the peak of the NSS distribution. Quantitatively, the mean of these 11 samples was 23.87, the standard deviation was small at 0.64; and the minimum and maximum values were 22.50 and 24.31. The mean of a corresponding random sample of ~ 5000 genotypes was 23.57 (standard deviation 1.48). That is, the NSS of these ribozyme SS are entirely typical of random structures of the given length. This is surprising if one considers the broad range of possible NSS in the full distribution, but perhaps less so given that nature appears to mainly use structures with large NSS.

Our finding may help rationalise the experimental result of Salehi-Astiani and Szostak [153] who found that when selecting random RNA for a given self-cleaving activity, the hammerhead ribozyme appeared most commonly. These findings also relate to the works of Knight *et al* [105, 104] who predicted that a surprisingly small number ($\sim 10^{10}$) of random RNA sequences should be enough to give rise to active hammerhead ribozymes and the isoleucine aptamer (tRNA).

Another important and well studied RNA molecule is the tRNA aptamer. We sampled 100 randomly chosen $L = 70$ tRNA molecules from the fRNAdb database [99]. The mean $\log_{10}$(NSS) was 29.80, the standard deviation was again small at 1.22 the minimum and maximum values were 26.64 and 32.62. The mean of a corresponding

random sample of $\sim$ 5000 genotypes was 29.40 (standard deviation 1.59). That is, the NSS of these SS are entirely typical of random structures of the given length.

3.3 Discussion

Our main findings in this chapter are:

(a) That the distribution of sequences over RNA secondary structures is well approximated by a simple log-binomial form, which facilitates the quantification of several important properties of the GP map such as the typical and total number of designable secondary structures;

(b) The distribution of NSS for random sequences is strongly peaked such that nearly all sequences adopt structures within a narrow range of NSS;

(c) That natural functional RNA from the RNAdb database [99] have SS for which the number of sequences per structure very closely resemble the distribution that would result from randomly sampling over genotypes; and further

(d) That the hammerhead ribozyme and tRNA motifs have among the most typical NSS, when compared to a random sampling over genotypes.

In relation to (a), it is worth pointing out that over the years significant effort has been put into determining N_P from combinatorial and computational analysis where either non-designable structures are counted or reduced sequence alphabet sizes have been used. Here we give an estimate of N_P based on a full alphabet, only count designable structures, and use large lengths L. Further, we defined (as in Chapter 2) a bias (β) ratio that allows us to determine a reduced number of states βN_P, which we argue take up the majority of genotypes. The value of N_P may therefore have less biological importance, as the majority of the phenotype space is effectively out of reach due to the bias, whereas the number βN_P gives a truer indication of the number of structures that are evolutionarily accessible to biological systems.

Perhaps the most significant of our findings is (b), that despite the astronomically large number of genotypes and phenotypes in RNA space, the mathematical structure of the GP strongly biases outcomes to a small region of phenotype space. One striking consequence of this narrow distribution in NS sizes is that it is possible to estimate the effective number of phenotypes (2^H) by using Eq. (3.19) and estimating $\bar{S}$, from only a few samples. It would be interesting to investigate in future work if this property is observed in other GP maps with bias. The previous chapter certainly suggests that this may very well be the case.

Another very significant finding is (c), namely that RNA SS in nature appear to closely follow the distribution predicted by a random sampling of genotypes. The results for the two iconic ribozymes, (d), are particular examples of the more general point (c). They illustrate the importance of NSS as an influential parameter in evolutionary dynamics: The strong similarity of the natural to the random NSS distributions suggests that NSS is not only important in that typically only frequent phenotypes will appear in nature (as stated by other authors earlier), but that NSS is a sufficiently influential quantity that natural RNA abundance is finely modulated by genotype-set sizes. We discuss this point further below.

Our findings build on other surprising similarities between natural and random RNA SS: Fontana *et al* [63] compared a small sample (34 sequences) of natural RNA SS to randomly sampled structures, and found that the structural features (e.g. distribution of number of bonds and loop sizes) of the natural and random SS were very similar. Later, Schultes *et al* [157] found that natural and random RNA are surprisingly similar in bond number and stem length. In this and a later experimental study [158] they also compared SS thermal stability, and found that for this property, natural RNA are notably more stable. Additionally, Smit *et al* [165] found that natural and random rRNA share strong similarities in the sequence nucleotide composition of SS motifs such as stems, loops, and bulges.

While the database could have various biases at work influencing which SS are present, it seems unlikely that these biases could account for the similarity between random SS and the database SS. It is true that if the folding package we use was

poor, then our results may be simply an artefact of a poor folding algorithm, and have little to do with properties of a real RNA. However, this seems unlikely as the Vienna package is extensively used and well tested. The NSS estimator could be at fault, but this also is well tested. Clearly, another explanation is needed for the correspondence we see.

Mutational robustness and evolvability are linked to NSS [184, 147, 91, 155, 80, 57]: Phenotypes with large NS sizes generally are more robust to mutations, and it has been argued that they are also more evolvable. Robustness is a selectable trait [189, 154], as too is evolvability [192, 137]. Thus one could argue that these factors contribute to understanding the overrepresentation of large NSS phenotypes in nature. Having said that, mere overrepresentation is not the same as the close match of natural distributions to random structures.

Another perspective is that possibly RNA SS is often not essential to RNA function, and hence under little selection, thus similar to a random sample. While in many cases SS is flexible (such as some parts of RNA virus [186]), it seems too strong to say that SS is largely irrelevant for all these RNA we examined. Indeed recent evidence points to SS as a strong determiner of tertiary structure [7, 8], and experimental work on aptamers have found SS to influence activity [21]. Additionally, recent texts state that SS is said to be very important for function in hammerhead ribozyme [50] and tRNA [127]. Finally, a recent study [144] found that "strong restrictions [are] imposed on the evolution of nc[non-coding]RNA molecules", a claim which does not seem to easily fit with idea that SS are largely freely evolving. See the Appendix to this chapter for more detail on the RNA used in our fRNAdb datasets, and the relevance of their SS for their functions.

One important point to keep in mind (given originally in the context of proteins) is that because the space of sequences is vast, evolution cannot have searched thoroughly through the space of genotypes for optimal solutions, and hence some contingency and randomness can be expected in phenotype outcomes [158]. Given the very large number of SS for the lengths we consider (e.g. $L = 55$, $N_P \sim 10^{13}$), this argument is relevant.

Recently it has been argued in detail that evolutionary dynamics strongly favours the "arrival of the frequent" [156], that is the rate at which variation appears is proportional to the NSS, and because these rates vary over so many orders of magnitude, this can heavily influence evolutionary outcomes. Thus the answer for why RNA follows the distribution predicted by random genotype sampling may well be that in many cases several different RNA SS will perform a given function roughly equally well (cf. [153, 182]). However, the SS with the largest NSS is most likely to appear as potential variation, and therefore the most likely to fix. Nevertheless, the close fit between predicted and observed NSS distributions is striking, and not fully explained.

Finally, we can return to a somewhat similar example, discussed in the previous chapter, namely the origin and evolution of the genetic code, which is a well studied but open question [107, 37, 42, 102]. One of the issues in this well studied problem concerns the observation of a positive correlation between the amino acid NSS sizes (i.e. number of associated codons) and their abundance in natural genomes [123, 100, 69, 85] (see also previous chapter). However, it is not clear whether the amino acid NSS sizes have adapted to organism amino acids requirements [123], or the NSS have a causal role in determining natural amino acid abundances [187, 103, 100], or some other explanation is responsible. In the previous Chapter, we inclined to the view of a causal influence of amino acid NSS; we suggest now that the results of this Chapter make more plausible the view that amino acid NSS plays a role in determining their abundances.

Future work should examine the extent, cause and quantitative distributions of phenotypic bias in other GP maps, and importantly how this bias interacts with selection in shaping evolutionary dynamics and outcomes. One aspect of this research direction would be to develop other means for accurately estimating the NSS of a given phenotype (as the NNSE we use here is not computationally efficient for very long RNA). We propose one method to do this in the next chapter.

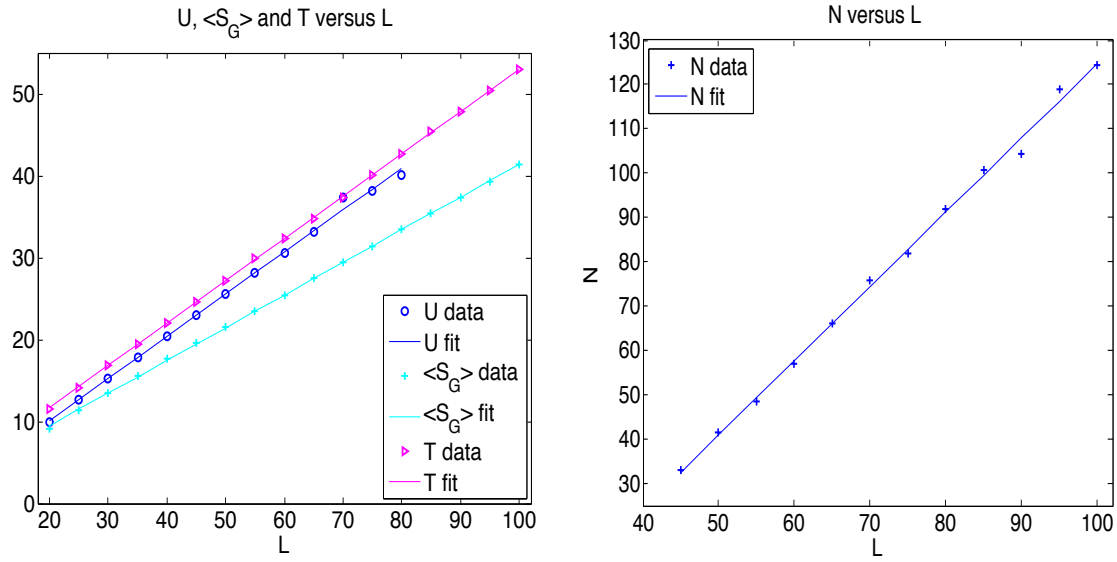
3.4 Methods

3.4.1 RNA secondary structure folding and neutral set size estimation

To fold a sequence to a SS, we use the `fold`-routine of the Vienna package pioneered by Schuster and colleagues [81]. All folding was performed with parameters set to their default values (in particular, the temperature is set at $T = 37°\mathrm{C}$).

The full enumeration of the $4^{20} \approx 10^{12}$ sequences for $L = 20$ that folded into 11219 took about 1 year of CPU time, and was performed by Schaper and Louis [156].

To determine the NSS for a given SS we used the neutral network size estimator (NNSE) described in Ref. [91] which uses sampling techniques. We used the default settings except for the total number of measurements (set with the -m option) which we set to 1 instead of the default 10, for the sake of speed, but this does not affect the outcome much.



(a) (b)

Figure 3.9: Some distribution parameters and number of SS (a) The maximum log NSS (excluding the trivial structure), U; the trivial structure's NSS; the mean ($\log_{10}$) NSS from sampling, $\bar{S}_G$. (Note that $< S_G >$ in the legend denotes $\bar{S}_G$.) (b) The fitting parameter N grows linearly with L.

3.4.2 Generating NSS distributions from sampling

In order to determine the sampled distributions shown for example in Figures 3.3 and 3.5 we use the following methodology: A (Python) random number generator was used to generate sequences, and the NNSE was used to find the size of the NSS. If the trivial structure was ever generated, the sequence was discarded. A small fraction of sequences were also discarded due to the NNSE failing to calculate the NSS. To generate a distribution, we partition the support of the distribution $[0, U]$ into bins. From this sampling method, we can then find the probability mass $P_G(S)$ in each bin.

In this work the sampling performed was as follows: For $L = 30$ we made 10^5 samples; for $L = 40$ we made 3×10^5 samples; for $L = 35, 45, 50, 55, 60, 65, 70, 75$ and 80 we made 20000 samples; and for $L = 85, 90, 95$ and 100 we made 5000 samples. (Steffen Schaper generated most of these data).

We can infer part of the P_P distribution from the same genotype sampled data. We do this as follows: Consider a bin $b_j \subset [0, U]$ of small width. We make the approximation that all structures within b_j have the same NSS, i.e. $10^{\tilde{b_j}}$, where $\tilde{b_j}$ is the value of the middle of the bin. If Q random genotypes are sampled, and q fall into the bin b_j, then we can infer

$$P_G(S = \tilde{b_j}) \approx \frac{q}{Q} \approx N_{P,j} \frac{10^{\tilde{b_j}}}{4^L} \tag{3.25}$$

where $N_{P,j}$ is the number of unique structures in bin b_j. Having values for $N_{P,j}$ for a range of bins gives an unnormalised estimate of distribution for P_P in those bins. Similar estimates for P_P can be made for other areas of the support for which sufficient sampling is achieved. We normalise the distribution by N_P, using the (independent) estimate of N_P as above. Hence we can infer

$$P_P(S = \tilde{b_j}) \approx \frac{N_{P,j}}{N_P} \tag{3.26}$$

In practice, choosing the bin sizes is a trade-off between being small enough to obtain a good description of the distribution, and large enough that the bins can be well populated with the samples sizes we use (which is important to reduce statistical fluctuations).

Finally, when plotting these estimates (purple diamonds on our graphs), very poorly populated bins with bin frequencies of ≤ 6 (being subject to large statistical errors) did not have associated data-points plotted. This is because if a bin contains only a few data points, then the error bars are so large that to plot corresponding frequency is not very informative. Additionally, from the point of view of data presentations, as our plots contain several distributions, the very large error bars would obscure parts of the other distributions. As we made 5000—20000 samples to make these plots, ignoring such poorly populated bins is reasonable.

3.4.3 Generating NSS distributions from database sequences

To generate the distributions of natural RNA we took all available sequences of $L = 20, 40, 50, 55, 60, 65, 70$ from the non-coding functional RNA database (fRNAdb [99]), and used the NNSE to estimate the NSS. These were then binned by NSS as above. Error bars in the plots were then taken to be simply statistical ($+/-$ square-root of bin frequency).

A small fraction ($\sim$1%) of the natural RNA sequences contained non-standard nucleotide letters, e.g. 'N' or 'R'; such sequences were ignored, because the standard packages cannot treat them. Similarly, a small fraction ($\sim$2%) of sequences were also discarded due to the NSS estimator failing to calculate the NSS. We also checked that there were no repeated sequences in the database of natural RNA. We assume that the probability of having repeated sequences by chance is so small that if there were repeated sequences, they would be due to mistakenly logging the same sequence. However, the database appears to be correct in this regard.

3.4.4 Estimating distribution parameters from sampling

To estimate the exponent of the NSS of the largest structure U, we sample and fold several thousand genotypes, using the NNSE to infer NSS. For smaller L we find repeats of the largest (non-trivial) NSS, but as the length increases both the time to infer the NSS and the number of samples needed go up. However, as the number of structures also increases, one can find many structures close to the absolute largest

U. We used samples for $L = 20, 25, ..., 75, 80$ to infer U. As can be seen from Figure 3.9, the dependence of U on L is very close to linear.

A very rough estimate of the error can be made for U found by 20000 samples with $L = 80$: Assuming that the maximum NSS U' we found in 20000 samples is not very much smaller than the true value of U, we can use U' to infer the probability that the maximum value from 20000 samples is within a small distance δU of U. This can be done by using U' to define a binomial distribution for S, and then estimating the probability that in 20000 samples, a SS with $S \in [U' - \delta U, U']$ is found. For $L = 80$, $U' = 40.02$, and we find that with $\sim 50\%$ chance, our estimate is within $\delta U = 1.05$ of U; and with $\sim 90\%$ chance, the estimate is within $\delta U = 1.35$ of U. Hence we can be reasonably sure that we have found a decent estimate of U for $L = 80$. Because finding U becomes easier with decreasing L, we can also be quite sure our estimates for U for $L < 80$ are good. As a small check, we also made an error estimate for U by using data for $L = 60$ ($U' = 30.56$) where the estimate of U will be very close ($\delta U = 0.25$ and $\delta U = 0.45$ with probability $\sim 50\%$ and $\sim 90\%$, respectively), but the shorter range of lengths makes fitting less reliable. Reassuringly, in this case we found a scaling of U extremely close to the one for $L = 80$, i.e. a scaling of $U \sim 0.513L$ instead of $\sim 0.514L$ (as in Eq. (3.6)).

To find T, we simply ran the NNSE for the trivial structures for $L = 20, 25, ..., 95, 100$. The scaling is very close to linear, and also quite close to U, as can be seen in Figure 3.9.

$\bar{S}_G$ was estimated by simply taking the mean S value from samples. The range was L =20, 25,...,95, 100. See Figure 3.9. From these we can obtain accurate linear fits.

Using our fit to $\bar{S}_G$, we can find p_G (and hence p_P) by the relation

$$p_G = \frac{\bar{S}_G}{U} \approx \frac{0.399L + 1.48}{0.513L - 0.2} \tag{3.27}$$

which implies that $p_G \to 0.7778 \approx 0.78$ from above, for large L. This implies, by Eq. (3.5), that $p_P \to 0.63$ for large L. These asymptotic values are approached quite

quickly, as even for $L = 70$, these limiting values are nearly reached: With $L = 70$, we have $p_G = 0.82$ and $p_P = 0.60$.

We use the measured mean $\bar{S}_G$ and variance σ_G^2 of the sampled P_G distribution to obtain N via

$$N = \frac{p_G(1-p_G)U^2}{\sigma_G^2} = \frac{\bar{S}_G(U-\bar{S}_G)}{\sigma_G^2} \tag{3.28}$$

Note that N need not be an integer. It is necessary to perform substantial sampling to get a good estimate of N for larger lengths, as the dependence on the variance σ_G^2 is quite sensitive. Further, as the sample variance is sensitive to the non-binomial behaviour we mentioned for $L \lesssim 40$, we only fit for N with samples for $L \geq 45$. Additionally, for smaller L the non-negligible probability of the trivial structure may affect calculations. Figure 3.9(b) shows N growing linearly for $L \geq 45$, with fit of

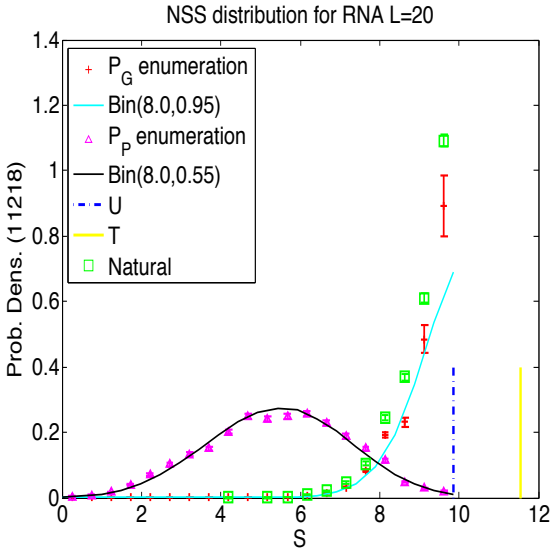
$$N = (1.68 \pm 0.03)L - (43 \pm 2) \tag{3.29}$$

The linear growth is clear, but difficult to say if N continues in a linear fashion for much larger L. While we did not fit to these estimates of N, for completeness we give some estimates for smaller L fixed by hand: Namely, for $L = 25, 30, 35, 40$ we have $N = 17.2$, 18.6, 22.8 and 26.9.

3.5 Appendix: Examining the fRNAdb datasets

In this Appendix, we look in some detail at the contents of some of the natural datasets we examined in the main chapter. In particular, we are interested in to what extent the RNA examined have functionally relevant secondary structures (SS). This is important to consider, because if, say, the majority of RNA in our datasets had structures that are irrelevant to function, then we would not expect selection to act on the SS, and hence the similarity between random and natural RNA would be unremarkable. On the other hand, if the RNA we study have structures which are important for their respective functions, then we should expect the structures to show signs of natural selection, and be different from random RNA.

We select a subset of the datasets to examine.



(a)

Figure 3.10: Natural $L = 20$ data (11896 structures) of RNA, excluding those labelled 'putative'. The natural data follows the distribution obtained from complete enumeration very closely.

3.5.1 $L = 20$

Figure 3.7 displays data for (approx. 14350) RNA of length 20 in the fRNAdb. The specific RNA and their corresponding proportions as a fraction of the total datasets are:

17 % Putative conserved noncoding region (EvoFold)
4.3 % Piwi-interacting RNA (piRNA)
1.2 % Mature microRNA
77 % Fly small RNA

The vast majority of these sequences came from the fruit fly *Drosophila melanogaster*. It is not clear to what extent the SS of these RNA are functionally relevant, and it may be that the structures are not relevant. Indeed, for the piRNA at least, it is not thought to have a known or conserved SS. Nonetheless, we have plotted the distribution for $L = 20$ for completeness, as it is the only length for which we have the complete NSS dataset.

Finally, we also plot the same $L = 20$ natural data, but with all RNA labelled 'putative' removed; see Figure 3.10. The comparison of natural and random RNA is quite striking in this new figure.

3.5.2 $L = 40$

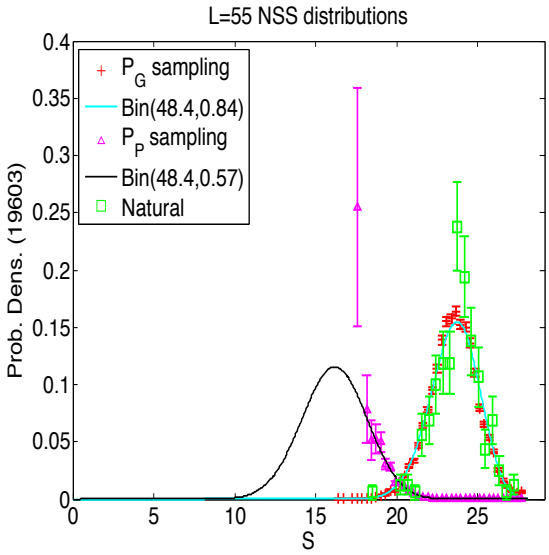
We now show the contents of the fRNAdb file for $L = 40$ with the approximate percentages:

0.3 % small non-messenger RNA (snmRNA)
0.3 % non-protein coding (noncoding) transcript
0.2 % Coronavirus 3' stem-loop II-like motif (s2m)
0.2 % Piwi-interacting RNA (piRNA)
74.5 % Putative conserved noncoding region (EvoFold)
0.6 % guide RNA (gRNA)
0.5 % HIV Ribosomal frameshift signal
22.2 % Prion pseudoknot
0.8 % U7 small nuclear RNA
0.3 % Ribosomal protein L19 leader
0.3 % Trans-activation response element (TAR)

We see that the majority (75%) are putative RNA, so it is not known whether the sequences are in fact real RNA. Hence these sequences may or may not have functionally relevant structures. The next largest fraction (22%) is absorbed by prion pseudoknots. It is debatable whether or not these prion RNA should be included in the NSS distribution plots, because these RNA contain pseudoknots, while the computational package we use does not handle pseudoknots. On the other hand, presumably the majority of a prion pseudoknot sequence is involved in specifying other standard SS motifs, and from this perspective it is still interesting to see how they compare to random RNA. Aside from the putative and psudoknot RNA, there are only $\sim$20 RNA sequences. In sum, we have plotted the data for $L = 40$ in the chapter main text, but it is not clear what we can learn from the plot.

3.6 $L = 55$

We now show the contents of the fRNAdb file for $L = 55$ with the approximate percentages:



(a)

Figure 3.11: Natural $L = 55$ data (213 structures) of RNA from the fRNAdb where RNA labelled 'putative' have been removed. The natural data follows the distribution obtained from random sampling closely, and is similar to the distribution plotted in Figure 3.5(d) which included RNA labelled 'putative'.

0.4 % self-splicing ribozyme RNA
1.2 % non-protein coding (noncoding) transcript
0.4 % HIV gag stem loop 3 (GSL3)
0.2 % small nucleolar RNA (snoRNA) SNORD82 / SNORD83A / SNORD83B / U82 / U83A / U83B / Z25
0.2 % Japanese encephalitis virus (JEV) hairpin structure
0.2 % precursor micro RNA (miRNA) mir-BART2
0.8 % guide RNA (gRNA)
16.9 % Unagi (eel) L2 (UnaL2) LINE 3' element
3.7 % small nucleolar RNA (snoRNA)
1.0 % Simian virus 40 late polyadenylation signal (SVLPA)
0.2 % small nuclear RNA (snRNA)
0.6 % Pyrococcus C/D box guide small nucleolar RNA (snoRNA)
40.9 % Putative conserved noncoding region (EvoFold)
0.2 % C/D box guide small nucleolar RNA (snoRNA) HBII-240 / SNORD72
16.3 % Putative conserved noncoding region (RNAz)
0.2 % C/D box guide small nucleolar RNA (snoRNA) HBII-210 / SNORD69
0.2 % C/D box small nucleolar RNA (snoRNA) HBII-429 / SNORD100
2.4 % Trans-activation response element (TAR)
0.2 % No description given
1.0 % small non-messenger RNA (snmRNA)
2.2 % Hammerhead ribozyme (type III)
0.2 % C/D box guide small nucleolar RNA (snoRNA) Z266
0.2 % small nucleolar RNA (snoRNA) snoR185
0.2 % predicted precursor micro RNA (miRNA) HP-67 [false negative]
0.2 % Rous sarcoma virus (RSV) primer binding site (PBS)
0.4 % Ribosomal protein L13 leader
6.5 % HIV Ribosomal frameshift signal
0.8 % C/D box small nucleolar RNA (snoRNA)
0.2 % putative archaeal H/ACA box small nucleolar RNA (snoRNA) u-46
1.2 % Group II intron
0.2 % C/D box guide small nucleolar RNA (snoRNA) HBII-239 / SNORD71
0.6 % Bovine leukaemia virus RNA packaging signal
0.2 % sok RNA

Looking at the largest contributors by percentage, we find that over half (41%, 16% and 0.2%) are labelled 'putative'. Because we do not know whether these sequences have functionally relevant structures, or indeed if they really are RNA, we also plot the $L = 55$ dataset with all putative RNA discarded (leaving 213 sequences remaining), see Figure 3.11. As can be seen, the similarity is of the natural to randomly sampled RNA is very close. Further, the distribution in this new figure is not notably different from the distribution which included putative RNA (Figure 3.5(d)).

Turning to the remaining non-putative RNA described above, 17% are eel LINE elements, which are known to have a stem-loop structure which is important for

function. Next most abundant (7%) are HIV ribsomal frameshift signal RNA, whose structure is important for interacting with the ribosome; snoRNA (4%) which has conserved and functionally relevant secondary structures; the Hammerhead ribozyme (2.2%) and Group 2 intron (1.2%) both of which are catalytic ribozymes and so have SS important for their functions. Further, by perusing over the other RNA with smaller abundance contributions, we see many such as Bovine leukaemia virus signal, precursor RNA, JEV hairpin structure, TAR and RSV which are known to have functionally relevant SS. In sum, for this dataset, at least the majority of the RNA have functionally relevant structures. Hence the similarity in terms of NSS of these natural RNA to random RNA is remarkable.

3.6.1 $L = 70$

Finally, we show the contents of the fRNAdb file for $L = 70$ with the approximate percentages:

0.0 % precursor micro RNA (miRNA) mir-99b
0.0 % small nucleolar RNA (snoRNA) Esmeraldo SLA1
0.9 % transfer RNA (tRNA), GCC (Gly/G) Glycine
0.3 % non-protein coding (noncoding) transcript
0.0 % precursor micro RNA (miRNA) mir-N367
0.7 % transfer RNA (tRNA), CAT (Met/M) Methionine
0.0 % precursor micro RNA (miRNA) mir-140
0.1 % C/D box guide small nucleolar RNA (snoRNA) SNORD38 / U38
0.0 % predicted precursor micro RNA (miRNA) RP-88 [false negative]
0.0 % precursor micro RNA (miRNA) mir-16c
0.0 % precursor micro RNA (miRNA) mir-24-1
0.0 % precursor micro RNA (miRNA) mir-303
0.0 % C/D box guide small nucleolar RNA (snoRNA) SNORD18A / U18A
0.0 % C/D box guide small nucleolar RNA (snoRNA) Z37
1.1 % transfer RNA (tRNA), TTC (Glu/E) Glutamic acid
5.1 % Putative conserved noncoding region (EvoFold)
0.0 % precursor micro RNA (miRNA) mir-K12-10b
0.0 % precursor micro RNA (miRNA) mir-200b
0.0 % transfer RNA (tRNA), GCG (Arg/R) Arginine
0.0 % C/D box small nucleolar RNA (snoRNA) HBII-429 / SNORD100
0.0 % precursor micro RNA (miRNA) mir-8
0.0 % precursor micro RNA (miRNA) mir-K12-3
0.0 % precursor micro RNA (miRNA) mir-199a-1
0.6 % No description given
0.1 % small non-messenger RNA (snmRNA)
0.2 % C/D box guide small nucleolar RNA (snoRNA) SNORD113 / SNORD114
0.0 % precursor micro RNA (miRNA) mir-196 / mir-196-1 / mir-196a-1

0.0 % predicted precursor micro RNA (miRNA) HP-27 [false negative]
0.1 % C/D box guide small nucleolar RNA (snoRNA) SNORD82 / U82 / Z25
0.0 % precursor micro RNA (miRNA) mir-412
0.0 % Plasmid R1162 RNA
2.8 % transfer RNA (tRNA), TCG (Arg/R) Arginine
0.0 % precursor micro RNA (miRNA) mir-23a
0.0 % precursor micro RNA (miRNA) mir-K12-8
0.0 % C/D box guide small nucleolar RNA (snoRNA) SNORD63 / U63
0.0 % precursor micro RNA (miRNA) mir-30 / mir-30d
0.0 % precursor micro RNA (miRNA) mir-US25-1
0.3 % precursor micro RNA (miRNA) lin-4
0.0 % predicted precursor micro RNA (miRNA) MP-64 [false negative]
0.0 % C/D box guide small nucleolar RNA (snoRNA) Z40
0.1 % transfer RNA (tRNA), GGT (Thr/T) Threonine
0.0 % precursor micro RNA (miRNA) mir-10b-2
14.4 % transfer RNA (tRNA), TGG (Pro/P) Proline
0.0 % C/D box guide small nucleolar RNA (snoRNA) SNORD2 / snR39B
0.0 % predicted precursor micro RNA (miRNA) HP-61 [all confirmed]
0.0 % precursor micro RNA (miRNA) mir-128a
1.1 % transfer RNA (tRNA), TGC (Ala/A) Alanine
0.1 % Enterovirus 5' cloverleaf cis-acting replication element
0.0 % S-element
0.0 % C/D box guide small nucleolar RNA (snoRNA) R38
0.1 % transfer RNA (tRNA), CAA (Leu/L) Leucine
0.0 % small nucleolar RNA (snoRNA)
1.2 % transfer RNA (tRNA), TGA (Ser/S) Serine
0.0 % C/D box guide small nucleolar RNA (snoRNA) SNORD18B / U18B
0.0 % C/D box small nucleolar RNA (snoRNA) 14q(I-1) / SNORD113-1
0.0 % C/D box guide small nucleolar RNA (snoRNA) SNORD41 / U41
0.0 % precursor micro RNA (miRNA) mir-US33
3.7 % Putative conserved noncoding region (RNAz)
0.1 % Selenocysteine insertion sequence
0.0 % Tombusvirus internal replication element (IRE)
0.0 % precursor micro RNA (miRNA) mir-10b-1
0.0 % C/D box guide small nucleolar RNA (snoRNA) SNORD34 / U34
0.0 % C/D box guide small nucleolar RNA (snoRNA) SNORD51 / U51
0.3 % HIV primer binding site (PBS)
0.0 % suhB
0.5 % HIV gag stem loop 3 (GSL3)
0.0 % precursor micro RNA (miRNA) mir-219
1.2 % transfer RNA (tRNA), GAA (Phe/F) Phenylalanine
0.2 % C/D box small nucleolar RNA (snoRNA)
0.1 % C/D box guide small nucleolar RNA (snoRNA) SNORD77 / U77
0.0 % sar RNA
3.6 % Group II intron
0.0 % C/D box guide small nucleolar RNA (snoRNA) Me28S-Cm788
0.1 % transfer RNA (tRNA), TCT (Arg/R) Arginine
0.1 % Vimentin 3' untraslated region (UTR) protein-binding region
1.3 % transfer RNA (tRNA), GAT (Ile/I) Isoleucine
0.0 % transfer RNA (tRNA), AGA (Ser/S) Serine
0.1 % precursor micro RNA (miRNA) mir-383
0.0 % precursor micro RNA (miRNA) mir-M30
0.1 % C/D box guide small nucleolar RNA (snoRNA) SNORD30 / U30
0.0 % small nucleolar RNA (snoRNA) Sylvio X10 SLA1

0.0 % sroB RNA
0.0 % small nuclear RNA (snRNA)
0.0 % transfer RNA (tRNA), CAC (Val/V) Valine
0.2 % C/D box guide small nucleolar RNA (snoRNA) SNORD18 / U18
6.8 % Selenocysteine transfer RNA (tRNA), TCA
0.1 % precursor micro RNA (miRNA) mir-30
0.0 % precursor micro RNA (miRNA) mir-K12-10a
0.2 % C/D box guide small nucleolar RNA (snoRNA) U2-30
0.1 % precursor micro RNA (miRNA) mir-BART1
0.0 % predicted precursor micro RNA (miRNA) RP-104 [false negative]
0.3 % RNA-OUT
0.0 % transfer RNA (tRNA), GGC (Ala/A) Alanine
0.0 % transfer RNA (tRNA), AGC (Ala/A) Alanine
10.8 % transfer RNA (tRNA), GTA (Tyr/Y) Tyrosine
0.2 % Rous sarcoma virus (RSV) primer binding site (PBS)
0.0 % C/D box guide small nucleolar RNA (snoRNA) SNORD58 / U58
4.9 % transfer RNA (tRNA), TAC (Val/V) Valine
0.0 % precursor micro RNA (miRNA) mir-694
0.1 % C/D box guide small nucleolar RNA (snoRNA) SNORD24 / U24
0.1 % precursor micro RNA (miRNA) mir-145
0.0 % C/D box guide small nucleolar RNA (snoRNA) SNORD56 / U56
0.9 % Tombus virus defective interfering (DI) RNA region 3
0.0 % precursor micro RNA (miRNA) mir-30d 0.4 % Flavivirus DB element
0.1 % precursor micro RNA (miRNA) mir-361
10.1 % transfer RNA (tRNA), TTG (Gln/Q) Glutamine
0.0 % transfer RNA (tRNA), GGG (Pro/P) Proline
0.3 % transfer RNA (tRNA), GTT (Asn/N) Asparagine
0.0 % transfer RNA (tRNA), GCT (Ser/S) Serine
0.0 % precursor micro RNA (miRNA) mir-K12-4
0.7 % transfer RNA (tRNA), GTC (Asp/D) Aspartic acid
0.0 % transfer RNA (tRNA), AGG (Pro/P) Proline
3.3 % transfer RNA (tRNA), TAG (Leu/L) Leucine
0.0 % small nucleolar RNA (snoRNA) Trypanosoma brucei RNA 9 (TBR9)
0.0 % precursor micro RNA (miRNA) mir-125b
0.0 % transfer RNA (tRNA), ACC (Gly/G) Glycine
0.0 % precursor micro RNA (miRNA) mir-137
0.1 % transfer RNA (tRNA), CCC (Gly/G) Glycine
0.0 % C/D box guide small nucleolar RNA (snoRNA) SNORD73 / U73
0.0 % precursor micro RNA (miRNA) mir-127
0.0 % transfer RNA (tRNA), ACA (Cys/C) Cysteine
0.0 % precursor micro RNA (miRNA) mir-M3
0.0 % msr RNA
0.0 % guide RNA (gRNA)
0.3 % transfer RNA (tRNA), TAA (Leu/L) Leucine
0.0 % predicted precursor micro RNA (miRNA) HN-3 [false negative]
0.0 % Glycine riboswitch
0.0 % Y RNA
0.1 % precursor micro RNA (miRNA) mir-183
3.3 % transfer RNA (tRNA), TGT (Thr/T) Threonine
0.0 % precursor micro RNA (miRNA) mir-206
0.0 % small nucleolar RNA (snoRNA) SNORD21 / U21
0.3 % transfer RNA (tRNA) with undetermined isotype
0.1 % C/D box guide small nucleolar RNA (snoRNA) Z12
0.0 % precursor micro RNA (miRNA) mir-13b-2

0.0 % precursor micro RNA (miRNA) mir-K12-5
0.5 % transfer RNA (tRNA), CTT (Lys/K) Lysine
0.2 % precursor micro RNA (miRNA) mir-32
0.0 % small nucleolar RNA (snoRNA) CL Brenner SLA1
0.0 % precursor micro RNA (miRNA) mir-147-1 / mir-147-2
0.0 % precursor micro RNA (miRNA) mir-790
0.0 % precursor micro RNA (miRNA) mir-302a
0.1 % C/D box guide small nucleolar RNA (snoRNA) SNORD50 / SNORD50A / U50
0.1 % transfer RNA (tRNA), GCA (Cys/C) Cysteine
0.0 % precursor micro RNA (miRNA) mir-369
5.7 % transfer RNA (tRNA), GTG (His/H) Histidine
0.1 % transfer RNA (tRNA), CCA (Trp/W) Tryptophan
0.0 % small nucleolar RNA (snoRNA) SNORD50A / U50
0.3 % Retroviral Psi packaging element
1.0 % transfer RNA (tRNA), TTT (Lys/K) Lysine
0.3 % precursor micro RNA (miRNA) mir-10
3.4 % transfer RNA (tRNA), TCC (Gly/G) Glycine
0.0 % C/D box guide small nucleolar RNA (snoRNA) R160

Perusing the above dataset descriptions shows that the majorty of the $L = 70$ dataset comes from tRNA, which clearly has a functionally relevant structure. Additionally there are many precursor microRNA, Group 2 introns, and snoRNA, all of which have functionally relevant structures. Finally, this dataset contains a relatively small proportion ($\sim$9%) of 'putative' RNA. In sum, for the $L = 70$ the vast majority of RNA in the dataset have SS important for function.

Chapter 4

Probability and Complexity

We propose a method for predicting the neutral set size of a phenotype based on its structural complexity, using little or no knowledge of the details of the GP map which generated the phenotype. The method is based on an area of theoretical computer science known as algorithmic information theory *(AIT). We view GP maps as information processing systems, and then use results from AIT to predict some probabilistic properties of these maps. Our work advances the still infant field of applied AIT, which seeks to find real-world applications for AIT and Kolmogorov complexity.*

4.1 Introduction

In the preceding chapters, we have discussed the ubiquity and evolutionary importance of phenotypic bias. In order to extend our understanding of bias in GP maps, it is useful to have ways to predict the neutral set sizes (NSS) of phenotypes of a given GP map. Making such predictions would facilitate NSS analysis of more complex biological systems where — unlike RNA — we do not have access to detailed computational models of the GP map, and also systems where — like RNA — we do have models of the GP, but estimating NSS for large systems (e.g. $L \gg 100$) is computationally demanding.

In this Chapter we propose a method for predicting NSS (or the probability) of a phenotype based on its structural complexity, using little or no details of the actual GP map. The mathematical framework underlying the method is based on an

area of theoretical computer science known as *algorithmic information theory* (AIT) [167, 106, 23, 24]. We view GP maps as information processing systems, and then use results from AIT to predict some probabilistic properties of these maps. As NSS and the probability of generating a phenotype from a random genotype are directly related, this allows prediction of NSS.

We find that for a large class of GP maps, NSS (or probability) is modulated by, and hence partially predictable by knowing, phenotype complexity (where 'complexity' is understood in the in AIT sense). Specifically, in 'simple' maps with bias, phenotypes of high complexity will tend to have exponentially low probabilities whereas some phenotypes of low complexity will have much higher probabilities. We call this phenomenon *simplicity bias*, as the bias in the maps tends to favour lower complexity, or simple, phenotypes. Moreover, based on tools from AIT, we make (sometimes quite accurate) predictions about the phenotype probabilities.

Further, while our motivation is to predict phenotype NSS in GP maps, the results of this Chapter are applicable far more widely than just to GP maps, indeed they may be applicable to many input-to-output maps where output complexity can be meaningfully quantified. Hence in order to highlight the relevance of this work beyond GP maps, this chapter will be framed in the language of 'inputs' and 'outputs', rather than genotypes and phenotypes.

To proceed, we will study the map

$$\mathcal{A} : I \to O \tag{4.1}$$

where I is the set of possible inputs ('genotypes'), O is the set of outputs ('phenotypes'), and $\mathcal{A}$ is some map ('GP map') which generates outputs using the inputs. N_I denotes the number of inputs, and N_O the number of outputs. We assume the map shows ('phenotypic') bias, which implies that $N_I \gg N_O$.

The main question of this Chapter is: In this mapping, can we make a non-trivial *a priori* prediction of the probability $P(x)$ of generating $x \in O$, on choosing a random input? Note that NSS and $P(x)$ are related by

$$NSS(x) = P(x)N_I \propto P(x) \tag{4.2}$$

where $NSS(x)$ is the NSS of phenotype x. Therefore, predicting $P(x)$ is essentially equivalent to predicting the NSS of x.

We approach this prediction question by applying results derived from Levin's *coding theorem* [110], which relates $P(x)$ to the structural information content or complexity of x, and then make such predictions in a variety of maps, including many maps that we examined in Chapter 2. The word 'complexity' is used in many different settings with different meanings [130]. However, the (structural) information complexity used in the coding theorem is *Kolmogorov complexity* (K-complexity), $K(x)$, a well defined quantity from AIT, which we now review.

4.2 AIT

We now give some notation, definitions, and key results of AIT that we will be using in this Chapter. Most of these are taken from Ref. [117], which is a standard reference for AIT. Other good introductions can be found in Refs. [19] and [67].

Throughout this chapter and the next, log is given in base 2. $l(x)$ denotes the length of a binary string x in bits. We denote the set of all binary strings of length n by $\{0,1\}^n$; similarly all binary strings of length $\leq n$ is written as $\{0,1\}^{\leq n}$. The set of all possible binary strings is denoted $\{0,1\}^*$. For a string x, x^n represents n concatenated copies of the string x. For example, $0^4 = 0000$.

We follow standard usage of the notation $O(f(u))$ as in AIT and physics [151]: For a function $\zeta(u)$ we write $\zeta(u) = O(f(u))$ if for all $u \geq u_0$ (for some $u_0 > 0$) we have $|\zeta(u)| \leq \mu |f(u)|$ for some constant μ. For example, if $t(u) = 2$ or $t(u) = \sin(u)$, then we can write $t(u) = O(1)$.

A *Turing machine* [178] is an abstract generic computation device that computes binary string outputs from binary string input strings. A *universal Turing machine* (UTM) is a Turing machine that can simulate the behaviour of any other Turing machine. A programming language that can be used to implement a UTM is called *Turing complete*. Many common programming languages such as C and Python are Turing complete.

The claim that any informal set of instructions can be implemented by a UTM is known as *Church's thesis* [117]. This claim is widely believed and justifies the approach of AIT which frames all computational problems in the context of UTMs and binary strings. We stress however that this does not mean that AIT only applies to binary strings, but rather that AIT applies to any system of mathematical objects which can be *represented* as binary strings, even if only in principle. While at first the framing of all problems in terms of binary strings may seem unfamiliar and very restrictive, in fact it is a common practice in much of science. For example, in much of physics and computational biology, problems are simulated on computers which ultimately represent and compute quantities in binary string form.

If a program running on a Turing machine stops after a finite number of steps, the program is said to *halt*. A *computable* map (or function) is one that halts for all inputs.

Proving that a given map is Turing complete is often not straightforward, as it must be proven to have equal computational power to another UTM. However one simple test for proving that a map is *not* Turing complete is to establish that all input programs halt. By this criterion, we see that many maps of interest in science (such as our GP maps) are computable; for example, any RNA nucleotide sequence we present to the computational folding package used in the last Chapter will adopt *some* secondary structure (i.e. output), and not keep running indefinitely.

The historical development of AIT by Solomonoff [167], Kolmogorov [106] and Chaitin [23, 24] was motivated by attempts at quantifying the information content or randomness of individual objects such as binary strings or discrete geometries. Shannon's information theory is related to AIT [117], but differs in that Shannon did not attempt to measure the information content or complexity of individual sequences, rather his theory pertains to the information content of random processes. That is, whereas Shannon information measures the information content of probability *distributions*, in contrast AIT attempts to quantify the information content of individual *objects*.

In AIT, information content is defined in terms of the amount of information needed to describe or construct a given object. As an example of the AIT approach, consider the following binary strings

$$\begin{aligned} (01)^n &= 0101010101010101010101... \\ rand(1:n) &= 1001110101011100010111111... \end{aligned}$$

The first is intuitively 'simple' as the whole string can be described as "print '01' n times". Hence as the string conforms to a simple rule — and thus also a short description — the string is deemed simple. The second string is a typical randomly generated bit-string. Presumably there are no hidden patterns in this string and it does not conform to any simple rules or short description, and hence it is deemed complex. In sum, AIT characterises the information content, or *Kolmogorov complexity*, of an output x as the length of the shortest computer program that can generate x.

More formally, the (plain) *Kolmogorov complexity* or *algorithmic complexity* $C_U(x)$ of a binary string x is defined as

$$C_U(x) = \min_q \{l(q) : U(q) = x\} \tag{4.3}$$

where $l(q)$ is the length of a binary program q in bits, and U is a UTM. That is, $C_U(x)$ is the length of the shortest program over all programs that print x and then halt. It is well known that $C_U(x)$ is *incomputable* [117], meaning that even in principle there cannot exist an algorithm for finding the exact value of $C_U(x)$, given x. This can be established by reduction to the *halting problem*, which is the name given to the fact that in general it is not possible to decide whether a given arbitrary program will halt or not, without actually running it to check (and even then you could not be sure that it would not halt except by waiting an infinite amount of time) [117].

By the *invariance theorem*, $C_U(x)$ only depends on the choice of the UTM U up to an additive constant. That is, if U and V are both UTMs, then

$$|C_U(x) - C_V(x)| \leq c \tag{4.4}$$

where c is a constant independent of x, but depending on the choice of U and V. Hence the subscript is dropped and we speak of 'the' Kolmogorov complexity $C(x)$.

The *conditional complexity* of x given y, written $C(x|y)$, is the length of the shortest program that can generate x, given the string y.

A code is a *prefix-code* if the set of code words (i.e. programs) is prefix-free [117], that is, if q and r are both valid programs which can produce outputs, then in a prefix code q cannot form the first $l(q)$ bits (i.e. it cannot form a prefix) of program r, and vice versa. This implies that there is no need for a spacer or other symbol to mark the beginning and end of concatenated programs. Hence any string of bits can be unambiguously decoded into separate programs.

Having introduced the plain complexity $C(x)$, we now define the closely related *prefix complexity* $K_W(x)$ [110, 23]:

$$K_W(x) = \min_q \{l(q) : W(q) = x\} \tag{4.5}$$

where W is a prefix UTM i.e. W is a *self-delimiting machine* where the programs are prefix-free. As above we drop the subscript of W on $K_W(x)$ also, writing simply $K(x)$. $K(x)$ is also incomputable. The *conditional (prefix) complexity* of x given y, written $K(x|y)$, is the length of the shortest program that can generate x, given the string y.

While $K(x)$ differs from $C(x)$ in important technical ways, quantitatively they are in fact very close, as $K(x)$ is equal to $C(x)$ up to a term logarithmic in $C(x)$: It is known [117] that

$$\begin{aligned} C(x) &\leq K(x) && (4.6)\\ &\leq C(x) + \log(C(x)) + 2\log\log(C(x)) + O(1) && (4.7)\\ &\sim C(x) && (4.8) \end{aligned}$$

The *universal probability* [110, 167] of a string x is defined as

$$P_U(x) = \sum_{q:U(q)=x} 2^{-l(q)} \tag{4.9}$$

which is the probability that a prefix UTM outputs x when fed with a random program (e.g. generated by coin flips). The probability of choosing a particular program of length l is 2^{-l}, so the universal probability simply sums over the probability of all possible programs. Note that the programs q are assumed prefix-free, and so by *Kraft's inequality*[1], $\sum_x P_U(x)$ converges[2].

An important theorem for our work is the (algorithmic) *coding theorem*, which was established by Leonid Levin in 1974 [110]. The theorem connects the K-complexity $K(x)$ and the universal probability $P_U(x)$ as follows

$$2^{-K(x)} \leq P_U(x) \leq 2^{-K(x)+d} \tag{4.10}$$

where d is some constant independent of x, but possibly depending on the choice of UTM, U. The coding theorem can be expressed differently as

$$P_U(x) = 2^{-K(x)+O(1)} \tag{4.11}$$

Due to the asymptotic independence of $P_U(x)$ on U, the subscript U is conventionally dropped, and we just write $P(x)$. In essence this theorem says that the probability of a computer printing x when fed with a random program is largely determined by the K-complexity of x: Low complexity ('simple') outputs are highly probable, and high complexity outputs are exponentially less likely.

The fact that the lower bound $P(x) \geq 2^{-K(x)}$ holds is clear, as the summation in Eq. (4.9) contains the term $2^{-K(x)}$. This lower bound was in fact pointed out earlier by Solomonoff [167]. The contribution of Levin was to show that the upper bound $P(x) \leq 2^{-K(x)+d}$ also holds (for some constant d). This latter claim is neither obvious nor trivial in the UTM setting [110, 35, 117]. For example, for UTMs there are infinitely many programs for any output, and so *a priori* we might have had high complexity strings being highly probable, due to having very many long programs. For example, a complex output with $K(x) = 100$ (for some chosen UTM) could conceivably have had $P(x) = 2^{-10} \gg 2^{-100}$ (when sampling programs on that same

[1]The inequality simply states that if $\mathcal{F}$ is a set of binary prefix-free code words, then $\sum_{f \in \mathcal{F}} 2^{-l(f)} \leq 1$

[2]This is one reason why prefix-free codes are needed.

UTM), by having 2^{90} different programs of length 100 bits. However, the theorem shows that this is not possible.

While AIT is an established field, applying results to real-world scientific and engineering problems is much less well explored. This is largely due to some theoretical difficulties with K-complexity and AIT, which we explore later in Section 4.7. However, in this Chapter we adopt an experimental approach: We attempt to bypass these difficulties by making various assumptions and approximations, and then test the resulting predictions for a range of examples. Once we have tested our approximations on our examples, we return to the more abstract question of how AIT applies to concrete computable maps (which are not Turing complete).

4.3 Results for computable maps

4.3.1 The coding theorem and computable maps

As the coding theorem (Eq. (4.11)) implies strong bias in output probabilities for input-computation-output maps, it is tempting to suggest that the coding theorem pertains to and makes predictions about input-output maps like $\mathcal{A}$. However, the theorem cannot be applied directly to these maps, due to the fact that the theorem assumes the presence of UTMs, while many maps of interest (such as many of the GP maps examined in earlier Chapters) are not UTMs, rather they are computable maps (cf. Section 4.2). Nonetheless, a weaker form of the coding theorem which applies to computable maps does exist: Li and Vitányi [117] give as a corollary to the coding theorem that for computable maps

$$P(x) \leq 2^{K(\mathcal{P})-K(x)+O(1)} \tag{4.12}$$

where $K(\mathcal{P})$ is the K-complexity of the list

$$\mathcal{P} = \{(x_1, P(x_1)), \ldots, (x_{N_O}, P(x_{N_O}))\} \tag{4.13}$$

for output objects $x_1, \ldots, x_{N_O} \in O$. That is, $K(\mathcal{P})$ is the length of the shortest program to generate the probability distribution defined by the set $\mathcal{P}$, which is the set of pairs of outputs and their corresponding probabilities.

4.3.2 Derivation of an upper bound

Due to ease of analysis, we will work with a slightly altered version of this upper bound (Eq. (4.12)), which we derive now, and which we make extensive use of in this Chapter. Before we do so, some notation must be introduced: $K(\mathcal{A})$ is the length of the shortest program to enumerate all inputs and map them to outputs via Eq. (4.1). Note that (abusing notation slightly), we use the letter $\mathcal{A}$ to denote both the *function* Eq. (4.1), as well as to denote the *program* which generates all inputs, and then computes outputs via Eq. (4.1). $K(x|\mathcal{A})$ is the information (in bits) required to specify x, given the mapping rule system $\mathcal{A}$.

To proceed with the mentioned derivation, we follow a general method outlined in Ref. [35], which applies to any computable function. Consider the algorithm:
(i) Enumerate all inputs of $\mathcal{A}$
(ii) Map these inputs to their outputs according to the rules specifying the map $\mathcal{A}$
(iii) Print the resulting list $\mathcal{P}$ (Eq. (4.13)) of outputs and their probabilities (i.e. frequencies/N_I).

Now, it is well known [35] from information theory that given a discrete distribution, one can efficiently encode outputs using a Shannon-Fano-Elias (SFE) code, which consists of prefix-free code words $E(x)$ of length (in bits)

$$l(E(x)) = \left\lceil \log\left(\frac{1}{P(x)}\right)\right\rceil + 1 \tag{4.14}$$

where $\lceil\cdot\rceil$ denotes taking the integer part. In this manner, we have a method for assigning bit strings to outputs x. So, using a SFE code, and given the mapping $\mathcal{A}$, we can describe any output x using $l(E(x))+O(1)$ bits, where the $O(1)$ term accounts for the fixed program to generate the SFE code. As K-complexity gives the shortest possible description length (within $O(1)$ terms) for a given UTM, we must have that the K-complexity of a given output x is no larger than the SFE code description just

derived, i.e.

$$\begin{aligned} K(x|\mathcal{A}) &\leq l(E(x)) + O(1) && (4.15)\\ &= \log\left(\frac{1}{P(x)}\right) + O(1) && (4.16)\\ \Rightarrow P(x) &\leq 2^{-K(x|\mathcal{A})+O(1)} && (4.17) \end{aligned}$$

An advantage of this form is that we can often make estimates about $K(\mathcal{A})$ and $K(x|\mathcal{A})$ from knowing the details of the map, whereas $K(\mathcal{P})$ is harder to work with analytically, due to being a less easily conceived quantity. We will use Eq. (4.17) extensively in the rest of our work.

Note: In contrast to the the coding theorem proper for UTMs, in the computable map case we do not have a corresponding strong lower bound on $P(x)$. Nonetheless, we can show (see Section 4.9 for a derivation) that with high probability — that is, for most inputs — $\log(1/P(x))$ is close to $K(x|\mathcal{A})$. However, this does not necessarily mean that $\log(1/P(x))$ is close to $K(x|\mathcal{A})$ for most *outputs*.

4.3.3 Simplicity bias

4.3.3.1 Simple mappings

This form of the coding theorem (Eq. (4.17)) applies to effectively any real-world mapping $\mathcal{A}$, provided only that in principle it could be computed on a UTM. Therefore it is a very general statement, and in general the details of $\mathcal{A}$ may affect the probabilities $P(x)$. To see why, consider that due to some artefact of $\mathcal{A}$ we might have $K(x|\mathcal{A}) \ll K(x)$, which would allow x to have a much higher probability than it would otherwise have according to the coding theorem (as $2^{-K(x|\mathcal{A})} \gg 2^{-K(x)}$). Thus in general it would be necessary to know the details of the map $\mathcal{A}$ in order to make predictions about $P(x)$.

We will leave for future work the general case of making predictions about $P(x)$ for arbitrary maps $\mathcal{A}$. Instead here we consider one important special case of map, which is nonetheless relevant to many GP maps. The special case is what we will call *simple maps*, which have the property that $K(x|\mathcal{A}) \sim K(x)$. Hence for simple maps

Eq. (4.17) implies

$$P(x) \lesssim 2^{-K(x)+O(1)} \tag{4.18}$$

i.e. that intrinsic output complexity $K(x)$ modulates $P(x)$, and not some system specific bias or artefact of the map $\mathcal{A}$. We call this property *simplicity bias*, due to the overall probabilistic bias for simpler outputs.

Establishing the full general conditions under which $K(x|\mathcal{A}) \sim K(x)$ holds is not straightforward, and we leave this for future work. However, we give one *sufficient* condition: For typical members $x \in O$,

$$K(\mathcal{A}) \ll K(x) \tag{4.19}$$

To see why this is sufficient, observe that

$$\left.\begin{aligned} K(x) &\leq K(x|\mathcal{A}) + K(\mathcal{A}) + O(1) \\ K(x|\mathcal{A}) &\leq K(x) + O(1) \\ K(\mathcal{A}) &\ll K(x) + O(1) \end{aligned}\right\} \Rightarrow K(x) \approx K(x|\mathcal{A}) + O(1) \tag{4.20}$$

which in turn implies Eq. (4.18), as required. Note that the first two inequalities in the brace are standard AIT results, whereas the third defines the special case of simple maps that we consider (Eq. (4.19)).

Informally, for a map to be 'simple' means that the structural variation in outputs x is generated largely by the information in the input, with as little artefactual biasing from the mapping rule-set as possible. One way to assess this in a particular map is that if the map *itself* clearly determines some/many aspects of output structure irrespective of input choice, then any complexity in these aspects must be due to artefactual biasing of the map, and not the information contained in a given input.

Finally, regarding further conditions for simplicity bias, we expect simplicity bias to become clearer for larger complexity outputs, as the asymptotic regime, where $O(1)$ terms are negligible, is approached (see Section 4.7 for more on this). Additionally, we expect simplicity bias to be seen more clearly for biased maps in which

$$0 \ll N_I \ll N_O \tag{4.21}$$

due to reduced finite size effects/statistical fluctuations.

4.3.3.2 Example of an asymptotically simple map

We now give an example of a map where condition Eq. (4.19) is met asymptotically: In the previous chapters we looked at the Vienna package computational map of RNA sequences folding to secondary structures. This map shows strong bias. Specifying this mapping consists of defining a fixed folding algorithm determined by biophysical rules, along with the value for the length L of the RNA sequences. Hence, for this map $K(\mathcal{A}) \leq \log(L)+O(1)$. On the other hand, typical output structures have complexity $O(L)$, which we infer from the fact that $\log(N_O) = O(L)$ (cf. previous chapter), and hence that $K(x) = O(L)$, by Section 4.5.1 (below). Thus asymptotically, for typical outputs, $K(\mathcal{A}) \leq \log(L) + O(1) \ll O(L) = K(x)$, and so the RNA GP map is a simple map.

4.4 Approximations

4.4.1 Approximations to $K(x)$

Due to the incomputability of $K(x)$, in our work we use approximations to $K(x)$ of bitstrings x, denoted $\tilde{K}(x)$. Several authors have found that some approximations to $K(x)$ behave surprisingly well in a variety of real-world settings, i.e. as if the approximations were close to the true K-complexity values [152, 59, 116, 26, 32, 197, 181]. We will use several different complexity measures to approximate $K(x)$, but mainly we use a measure that we call $C_{LZ}(x)$, described now.

In 1976 an influential paper introducing a complexity measure for digital strings (or sequences) was published by Lempel and Ziv [109]. Their algorithm formed the foundation for some now-common compression algorithms. The essence of the algorithm is to read through a string (of any finite alphabet size) and create a dictionary of new sub-patterns as they appear in the string. Thus a string with many different sub-patterns would yield a large dictionary, hence being assigned a high complexity; while a string of little variation and essentially built up of repeated sub-patterns would yield a small dictionary, hence being assigned a low complexity. This complexity function is denoted $c(x)$. The function $c(x)$ was not originally intended as

an approximating function for K-complexity, rather only as one possible way to measure the complexity of a sequence. However, it has since been used to approximate K-complexity [92].

As it stands, $c(x)$ has a weakness as a K-complexity measure in that if $x = 0^n$ for any $n \geq 2$, then $c(x)$ returns a constant complexity value, independently of n; in contrast, $K(0^n) = O(\log(n))$. With this in mind, we used instead a simple transformation of $c(x)$ to make the complexity measure $C_{LZ}(x)$ as

$$C_{LZ}(x) = \begin{cases} \log(n), & \text{if } x = 0^n \text{ or } 1^n \\ \log(n)[c(x_1x_2...x_n) + c(x_n...x_2x_1)]/2, & \text{otherwise} \end{cases} \tag{4.22}$$

with $n = l(x)$. The reason for distinguishing 0^n and 1^n is merely an artefact of $c(x)$ which assigns complexity 1 to the string 0 or 1 but complexity 2 to 0^n or 1^n for $n \geq 2$ which is problematic in that this means $c(x)$ is inconsistently sensitive to string length. Taking the mean of the complexity of the forward and reversed string makes the measure more fine grained in the sense of having more different possible complexity values. Note that C_{LZ} can also be used for strings of larger alphabet sizes than just 0/1 binary alphabets.

A MATLAB file (written by Stephen Faul) for computing $c(x)$ for binary strings (only) is available at

`www.mathworks.co.uk/matlabcentral/fileexchange/6886-kolmogorov-complexity`

We wrote another MATLAB program which implements the original algorithm for any finite alphabet size.

Note: Strictly, the coding theorem results which we invoke apply to the prefix-free version of K-complexity, denoted $K(x)$, as opposed to the plain K-complexity, denoted $C(x)$ [117]. However, these two measures are asymptotically equal by Eq. (4.6), and essentially almost identical. Since we approximate $K(x)$ anyway using for example C_{LZ}, we ignore the subtle distinction between these measures in our approximations.

4.4.2 Approximation to the upper bound

A condition we stipulated for observing simplicity bias in biased maps is that $K(x|\mathcal{A})$ is closely linked to $K(x)$. Hence, to make progress, we introduce the simplfying

assumption that the exponent in Eq. (4.17) is related to our approximation $\tilde{K}(x)$ by

$$K(x|\mathcal{A}) + O(1) \approx a\tilde{K}(x) + b \tag{4.23}$$

for constants $a > 0$ and b. These constants account for the $O(1)$ term, potential idiosyncrasies of the complexity approximation $\tilde{K}$, and other possible factors arising from our approximations. Hence we approximate Eq. (4.17) as

$$P(x) \lesssim 2^{-a\tilde{K}(x)-b} \tag{4.24}$$

Note that the constants a and b depend on the mapping, but not on x.

As we discuss in the next Section and the example maps below, the values of a and b can often be inferred *a priori* using one or more of: The complexity values of all the outputs, the number of outputs (N_O), the probability of the simplest structure, or other values.

4.5 Making predictions for $P(x)$ in computable maps

We can often make predictions about the values of a and b (Eq. (4.24)), via various methods. Essentially we use any piece of information about the outputs or their probabilities that is available to estimate, bound or approximate the values of a and b. We now describe some methods, which we apply to various maps in the next Section of this Chapter.

4.5.1 Estimating the range of $K(x|\mathcal{A})$

We will now estimate the range of values that we expect $K(x|\mathcal{A})$ to assume. We begin with a lower bound on possible complexity values: Given $\mathcal{A}$ we can compute all the inputs, and produce all N_O outputs. Hence, we can describe any $x \in O$ by its index $1 \leq j \leq N_O$ in the set of outputs O. Therefore

$$K(x|\mathcal{A}) \leq \log(j) + O(\log(\log(j))) \tag{4.25}$$

where the second $O(\log(\log(j)))$ term arrises from the fact that the description is in prefix-free form. Now, if $\mathcal{A}$ is a simple map then the inequality just stated implies

$$K(x) \lesssim \log(j) + O(\log(\log(j))) \tag{4.26}$$

This implies that there must be some simple outputs, otherwise we could not describe them with small integer indexes of $j \approx 1$, which have low K-complexity: $K(1) \approx 0$. Therefore, the smallest complexity value is expected to be

$$\min_{x \in O} \left(K(x|\mathcal{A}) \right) \approx 0 \tag{4.27}$$

Next, we make an estimate of the *smallest* value that $\max(K(x|\mathcal{A}))$ could attain, for $x \in O$: If the set of programs for outputs x was the most efficient conceivable, and even ignoring the fact that programs should be prefix free, even so such a set would contain programs of length $\sim \log(N_O)$. To see this, notice that we must have enough programs for all N_O outputs, but there simply are not enough short programs to allow all outputs to have short programs[3]. Quantitatively, we can see this by enumerating all bit string programs (in ascending order of length), and estimating the minimum upper bound $\mathcal{L}$ of program lengths for there to be sufficient programs to map to all N_O different outputs. This estimate will be

$$\sum_{r=1}^{\mathcal{L}} 2^r = 2^{\mathcal{L}+1} - 1 \approx N_O \tag{4.28}$$

$$\Rightarrow \mathcal{L} \approx \log(N_O) \tag{4.29}$$

Hence, some programs must be at least $\sim \log(N_O)$ bits, i.e.

$$\max_{x \in O} \left(K(x|\mathcal{A}) \right) \geq \log(N_O) + O(1) \tag{4.30}$$

In fact not only 'some' programs, but most programs will be this length, because in the set of programs $\{0,1\}^{\lesssim \mathcal{L}}$, most programs have length $\sim \mathcal{L}$.

We now wish to derive the *largest* value that $\max(K(x|\mathcal{A}))$ could attain, for $x \in O$: Recall that given $\mathcal{A}$ we can compute all the inputs, and produce all N_O outputs. Therefore the information for N_O is contained in $\mathcal{A}$. Hence

$$K(x|\mathcal{A}) \leq K(x|N_O) + O(1) \tag{4.31}$$

[3]This argument is one of the most basic and well known results of AIT.

Further, for any $x \in O$, we could describe it by assigning and specifying a length $\lceil \log(N_O) \rceil$ bit string. This representation of all outputs as strings of $l(x) = \lceil \log(N_O) \rceil$ means that we can use the following known relation of $K(x|l(x))$ to plain complexity $C(x)$ [117]

$$\begin{aligned} K(x|\mathcal{A}) &\leq K(x|N_O) + O(1) && (4.32)\\ &= K(x|l(x)) + O(1) && (4.33)\\ &\leq C(x) + O(1) && (4.34)\\ &\leq l(x) + O(1) && (4.35)\\ &\leq \lceil \log(N_O) \rceil + O(1) && (4.36)\\ &\leq \log(N_O) + O(1) && (4.37) \end{aligned}$$

for any $x \in O$. Now, this upper bound and the lower bound of Eq. (4.30) coincide at $\log(N_O)$, and so the largest complexity value is expected to be

$$\max_{x \in O} \left(K(x|\mathcal{A}) \right) \approx \log(N_O) \tag{4.38}$$

In sum, the range of complexities is expected to be

$$0 \lesssim K(x|\mathcal{A}) \lesssim \log(N_O) \tag{4.39}$$

for all $x \in O$.

4.5.2 Estimates of the gradient a from N_O and $\max(\tilde{K}(x))$

In this Section we find a close connection between the gradient a of the decay in $\log(P(x))$ with complexity, and the total number of outputs, N_O. The connection is derived as follows: We have lower and upper bounds on $K(x|\mathcal{A})$ from Eq. (4.39). Hence, the minimum and maximum of the upper bound in Eq. (4.17) are, respectively

$$\min_{x \in O} \left(2^{-K(x|\mathcal{A}) + O(1)} \right) \sim \frac{2^{O(1)}}{N_O} \tag{4.40}$$

and

$$\max_{x \in O} \left(2^{-K(x|\mathcal{A}) + O(1)} \right) \sim 2^{O(1)} \tag{4.41}$$

We assume that this minimum upper bound is reached by the most complex objects, i.e. outputs such that $\tilde{K}(x) = \max_x(\tilde{K}(x))$. Putting these together means that the gradient a will be roughly

$$a \approx \frac{\log(2^{O(1)}/N_O) - \log(2^{O(1)})}{0 - \max_{x \in O}(\tilde{K}(x))} \tag{4.42}$$

$$\approx \frac{\log(N_O)}{\max_{x \in O}(\tilde{K}(x))} \tag{4.43}$$

This shows the connection between the gradient a and N_O. (As an example, in the case $O = \{0,1\}^n$, and $\tilde{K}(x) = K(x)$, then $\log(N_O) = n$ and $\max(x) = n$, so that the gradient would be $a = 1$.) Using the form for the gradient just derived implies that the minimum and maximum upper bounds in terms of the approximation will be $2^{-b}/N_O$ and 2^{-b}, respectively.

4.5.3 Estimating $\max(\tilde{K}(x))$ for a

In order to use Eq. (4.43) to estimate a, we need first to estimate $\max_x(\tilde{K}(x))$. If we have the entire collection of outputs available to analyse, then this estimate is trivial as we just compute $\tilde{K}(x)$ for each and then take the maximum. If we do not have the entire collection of outputs, then we may still be able to estimate the maximum complexity, using the form of the outputs. A simple example of such an estimate is the case where the outputs O are a subset of $\{0,1\}^n$, and so we know that

$$\max_{x \in O}(\tilde{K}(x)) \sim \max_{x \in O}(K(x)) \tag{4.44}$$

$$\lesssim \max_{x \in \{0,1\}^n}(K(x)) \tag{4.45}$$

$$\sim n \tag{4.46}$$

or whatever maximum complexity value our particular choice of approximation $\tilde{K}(x)$ returns for random strings of length n. Now, if $\log(N_O)$ is not very much smaller than n then it is reasonable to approximate $\max_{x \in O}(\tilde{K}(x)) = n$, or whatever maximum value of $\tilde{K}(x)$ returns for random strings of length n. Similarly, the lowest complexity value over all outputs, $\min_{x \in O}(\tilde{K}(x))$, can sometimes be estimated by knowing the form

of the objects. Keeping the same example from above of length n bit strings, then the lowest complexity would be $\sim \log(n)$, because it is known that $K(00...00) \sim \log(n)$.

4.5.4 Other ways of calculating a, $\max(\tilde{K}(x))$, and N_O

4.5.4.1 Using the gradient

Using Eq. (4.43) we can sometimes find the gradient from knowing N_O, or find N_O from knowing the gradient. In a particular case, we may already know N_O from previous work. Alternatively N_O may be found by thorough sampling of inputs or complete enumeration of inputs. Alternatively, if we know the gradient and $\max(\tilde{K}(x))$, we can infer N_O directly (using Eq. (4.43)). One advantage of deriving N_O from a is that a can sometimes be found from only relatively sparse sampling, i.e. only enough to read off the gradient from the relevant $\log P(x)$ vs $\tilde{K}(x)$ plot.

Note: As a practical point, when performing partial sampling, one must be careful not to include high complexity/low frequency outputs in the gradient estimation, as these low frequency outputs are likely to have large statistical errors. Including these will tend to underestimate the magnitude of the gradient.

4.5.4.2 Using the form of the outputs

We may also be able to bound N_O by knowing the form of the outputs. For example, RNA SS can be written in dot-bracket form, hence as there are three possible symbols per base, then $N_O \leq 3^L$. Similar estimates can be obtained for outputs such as subsets of binary strings. Having suggested this method, it may well lead to only weak upper bounds due to strongly overestimating via these simplistic combinatorial arguments. For example, with $L = 55$, we calculated in the last chapter that $N_O \sim 10^{13}$, but $3^{55} \sim 10^{26}$ is much larger.

4.5.5 Estimating b

We can predict b in same cases, as described below. As an important comment, we highlight that even if estimating b is difficult, knowing just a can be used to predict

whether, for example, $P(w) > P(x)$ with $w, x \in O$, which in many cases is nearly as valuable as estimates for the exact values of $P(w)$ or $P(x)$.

4.5.5.1 Sum of probabilities

If the entire collection of objects is available, then if a is known, then b may be estimated via the constraint that probabilities must sum to 1 (because we are analysing computable distributions, this allows us to avoid known normalising difficulties with semicomputable distributions [117]). Mathematically,

$$\begin{aligned} 1 &= \sum_{x \in O} P(x) && (4.47) \\ &\approx \sum_{x \in O} 2^{-a\tilde{K}(x)-b} && (4.48) \\ \Rightarrow b &\approx \log(\sum_{x \in O} 2^{-a\tilde{K}(x)}) && (4.49) \end{aligned}$$

Similarly, if b is known or can be estimated, then the same method could be used to find a. One problem with this approach is that in some maps, we have many outputs with probabilities well below their upper bounds. Now, as this approach assumes $P(x) \approx 2^{-a\tilde{K}(x)-b}$, then this will lead to inaccuracies in estimating a or b.

4.5.5.2 If N_O is small, then $b \approx 0$

If N_O is small, then it is reasonable to assume $b \approx 0$. The reason is as follows: The mean probability will always be $1/N_O$; also we expect the greatest upper bound (Eq. (4.41)) to be much larger than the mean, so that

$$\begin{aligned} 2^{-0\tilde{K}(x)-b} &= 2^{-b} \gg 1/N_O && (4.50) \\ \Rightarrow \log(N_O) &\gg b && (4.51) \end{aligned}$$

So, taking $N_O = 250$ for example, then $\log(N_O) \approx 8$, so assuming $b \approx 0$ is reasonable. Clearly this approximation will hold more closely the smaller N_O is.

4.5.5.3 Estimating b from a specific x and $P(x)$

If $P(x)$ is known for some output x, then assuming knowledge of a and $\tilde{K}(x)$, and assuming

$$P(x) \approx 2^{-a\tilde{K}(x)-b} \tag{4.52}$$

then b can be inferred by rearranging this equation. Clearly this method would work just as well for finding a, if you knew b and $\tilde{K}(x)$.

One drawback with this method is that it relies on the assumption of the approximate equality Eq. (4.52); hence if for the chosen output x, the approximation was only a very poor one, then the corresponding estimation of b would be equally poor. This comment is important as we will see that for many of the example maps that follow, many outputs have $P(x)$ far below their respective upper bounds, such that Eq. (4.52) would not hold well at all.

Having said that, we have noticed that in nearly all maps studied, the $P(x)$ values for very simple outputs are close to the predicted upper bound. Hence, in general, simpler outputs should be chosen for this method over more complex outputs. Conveniently, as these are typically among the most probable of all outputs, they are also relatively easy to find by sampling. So, in practice what we can sometimes do is sample many inputs, enough to obtain an estimate of $P(x)$ for the most likely output. Then we can use this knowledge in addition to some other information (like a or N_O), and thereby find b.

4.5.5.4 Calculating b from the mean complexity

Assuming knowledge of a, we can estimate b by using the mean sampled complexity, i.e.

$$\begin{aligned}
\langle\tilde{K}\rangle &= \sum_{x\in O} P(x)\tilde{K}(x) && (4.53)\\
&\approx \sum_{x\in O} 2^{-a\tilde{K}(x)-b}\tilde{K}(x) && (4.54)\\
\Rightarrow b &\approx \log(\sum_{x\in O} 2^{-a\tilde{K}(x)}\tilde{K}(x)) - \log(\langle\tilde{K}\rangle) && (4.55)
\end{aligned}$$

An advantage of this method is that estimating $\langle \tilde{K} \rangle$ from sampling can often be obtained with only relatively few samples inputs. However, the method does require having the entire set O, or at least enough of the elements of O to make a decent approximation of the right hand side of Eq. (4.54).

In the next Section we try a series of concrete example maps, and make predictions about $P(x)$ based on the range of methods just described. This excersise will be a good test for our general methodology and theory.

4.6 Example maps

4.6.1 Getting a lot from a little

In this Section we turn to example maps, and testing our *a priori* predictions for $P(x)$. Before doing so, we highlight two points that should be kept in mind while examining the predictions.

Firstly, while the predictions are not always precise, and in reality only form predictions for a *bound* on $P(x)$, we stress that these predictions are based on very little system specific knowledge. In general, the more one knows about a system, the more precisely one can predict its properties. In this Chapter, we try to make predictions about system properties (i.e. $P(x)$) using only minimal knowledge about the maps, and instead mainly use the complexities of the outputs that arise. As such, while system specific complexity measures or structural heuristic estimators of $P(x)$ may be constructed which yield more precise predictions than ours, it should be remembered that comparing to those such predictors would not be a fair comparison, if the system specific method assumed much more about the maps than our method does.

Secondly, these predictions are based on some coarse approximations, which may well be improved on in the future, and in turn may improve the predictions. Specifically, we make approximations to Kolmogorov complexity, use the simplified form of Eq. (4.23), and also make further approximations and estimations for the constants a and b in Eq. (4.23).

4.6.2 Maps

4.6.2.1 Polyominoes

Model: Polyominoes are a model computational model system for self-assembly, where 2D square blocks self-assemble via specified interactions between different block edges [1, 90, 72]. In this polyomino map, the inputs determine the interactions between blocks, and the outputs are 2D block assemblies of different sizes and shapes. An example of blocks self-assembling into a cross shape is given in Section 2.2.3.3. Polyominoes can be made using different parameter sets, which specify the number of tiles (N_t) and the number of colours (c) used; a given parameter set for the map is written $S_{N_t,c}$. The colours determine the number of possible interaction types between the tiles.

Methods: We used data for polyominoes with parameters $S_{2,8}$ and $S_{3,8}$. The data for the probabilities $P(x)$ from full enumeration and polyomino outputs (shapes) were taken from Refs. [90] and [72], respectively. (Sam Greenbury kindly provided the data).

The complexity measure we use for polyominoes is *path complexity*, a novel measure that we introduce here: To estimate the path complexity of a polyomino, we first pick an edge of the polyomino, then 'walk' around the full perimeter of the polyomino, and record whether each step was forward (F), i.e. in the same direction as the previous step, right (R), i.e. required stepping right with respect to the preceding direction, or left (L). Thus a shape gives rise to a string consisting of the letters F, R and L. In this manner, larger objects have longer string representations, and repeated structural motifs can generate repeated letter motifs in the string (e.g. FFL may appear often). Given this three-letter alphabet string, we estimate the string complexity using C_{LZ}. As the string will depend on the starting point of the walk around the polyomino perimeter, we take the minimum complexity value over all possible starting positions.

A drawback of our path complexity measure is that it is designed to work for shapes with no 'holes', such that specifying the perimeter entirely specifies the poly-

omino shape. For the data sets we used, this was not a significant problem, as only a small fraction of structures had holes (because the shapes were small). To calculate the complexity for these structures with holes, we added the complexity of the hole perimeter to the complexity of the exterior perimeter. The intuition for this addition is that the polyomino could be described by the exterior perimeter, and then cutting out the interior hole.

One important positive aspect of this path complexity measure is that it is not system specific: It assumes nothing about the way the polyominoes were made, but only measures the shape geometrical complexity. This is in contrast to another complexity measure for polyominoes [89] which is based on the number of different tiles required to make the polyomino, thus implicitly using knowledge that the shape arose from assembling blocks of different types.

Prediction: Figure 4.2(a) shows a plot of $P(x)$ vs. estimated shape complexity. The red line shows the prediction, which matches the data quite well. To make this prediction, we assumed that $b = 0$, which is justified by knowing that the number of objects $N_O \approx 150$, is small (see Section 4.5.5.2), and used it to find a (see Section 4.5.5.1). The values $a = 0.51$ and $b = 0$ were used.

Thus we made estimates of the probabilities $P(x)$ for the various output shapes almost directly from the shapes themselves, without assuming details about the self-assembly mapping that produced them. See Figure 4.3(a), Section 4.10, for a plot of the $S_{2,8}$ polyomino model.

4.6.2.2 RNA secondary structure

Model: The RNA nucleotide to secondary structure (SS) map is a well studied sequence-structure map [160] (See also Chapter 2 and 3 of this thesis). The inputs here are RNA nucleotide sequences, and the outputs are RNA SS, and the mapping is a computational algorithm for determining minimum free energy SS.

Methods: In our analysis, generating RNA sequences and structures and estimating $P(x)$ (from neutral set sizes) was performed as in Chapter 3.

To estimate the complexity of an RNA SS, we converted the dot-bracket representation of the structure into a binary string, and then used C_{LZ} to estimate the complexity of the resulting binary string. To convert to binary strings, we replaced each dot with the bits 00, each left-bracket with the bits 10, and each right-bracket with 01. Thus an RNA SS of length L becomes a bit-string of length $2L$. As an example, the following $L = 12$ structure yields the displayed 24-bit string

$$(((...)))... \rightarrow 101010000000010101000000$$

(Note: We could have used C_{LZ} directly with the three letter alphabet of dot, left-bracket, right-bracket, but we found the binary representation to be a marginally better predictor of $P(x)$).

Prediction: To make the predictions for $P(x)$ in Figure 4.2(b), firstly the gradient a was estimated via Eq. (4.43) by using our estimated values of N_O from the preceding Chapter, in addition to an estimated value for $\max(\tilde{K})$. For the latter quantity, we made the approximation that $\max(\tilde{K}) = C_{LZ}(\zeta_{2L})$, where ζ_{2L} is a random bit string of length $2L$. The reason for this approximation is to accommodate an artefact of C_{LZ} that for short strings C_{LZ} returns large complexity values, i.e. $C_{LZ} \gg 2L$. For longer strings or better complexity estimators which do not suffer from this artefact, we could simply make the approximation $\max(\tilde{K}) = 2L$. (Rather than estimating the complexity of a single random bit string, we took the largest complexity over 250 random bit-string samples). This estimation of $\max(\tilde{K})$ is an example of the method described in Section 4.5.3.

To estimate b, we used the calculated value of $P(x)$ for the trivial structure (with no bonds), and then, given that we had a and $\tilde{K}(x)$ (which is just $\log(2L)$), we used Eq. (4.52) as in Section 4.5.5.3. The values $a = 0.30$ and $b = 8.83$ were used.

The upper bound prediction in Figure 4.2(b) is good, though many SS have lower than predicted NS sizes. Notice that the gradient prediction is quite accurate, and that this was determined without knowing any details of the mapping process (except N_O), though we did use some system specific knowledge to fix b (i.e. $P(x)$ for the trivial structure).

Comment: In this RNA model, we found that coding theorem-like behaviour becomes more pronounced for longer L: In Figure 4.4 (Appendix) we show analogous plots to Figure 4.2(b) but with $L = 20$, 30 and 80. For $L = 20$ the decay in log probability with complexity and the correlation of these two quantities is modest, whereas for larger L such as 55 and 80 the correlations are more striking. At present it is not clear why the connnection of probability to complexity should be less pronounced for smaller system sizes, and this requires further investigation in the future.

4.6.2.3 Polynomial curves

Model: Consider the polynomial curve $y(t)$ of degree n defined by

$$y(t) = \alpha_n t^n + \alpha_{n-1} t^{n-1} + \cdots + \alpha_1 t + \alpha_0 \tag{4.56}$$

with $t \in [0, 1]$. The inputs here are the values α_i, $i = 0, 1, \ldots, n$, and the outputs are the different resulting curves $y(t)$, where curves are coarsely discretised to binary strings.

Methods: The curve discretisation follows the 'up-down' method [190, 60]: For discrete values of $t = \delta t, 2\delta t, 3\delta t \ldots$, if $dy/dt \geq 0$ at time $j\delta t$, then a 1 is assigned to position j of the binary string, and otherwise a 0 is assigned. In this case, we chose $\delta t = 0.02$, which yields a length 50 bit string output. The choice of the value for δt is a trade-off between being so small that outputs are so fined grained that estimating $P(x)$ from sampling is difficult (due to there being so many different outputs, that any given output appears with low frequency), and δt being so large that there are only a small number of different outputs which have a small range of complexity values (as the length of the binary string is inversely related to δt). To estimate discretised curve complexity, C_{LZ} was used with the binary string form of the curve as an argument.

The values of α_i were chosen randomly with distribution $\mathcal{N}(0, 1)$, but other choices or distribution were also made (see below). For each choice of sampling for α_i, 10^6 random samples were made, except for the $\alpha_i = \pm 1$ case (below) where full enumeration of inputs was performed. The polynomial degree was chosen to be $n = 14$.

Prediction 1: We approximated that $b = 0$. While we could not predict *a priori* that $b \approx 0$ (N_O was a little large to use the method Section 4.5.5.2), in this map it can be seen from partial sampling that some outputs have high probability, not far from 1, which shows that $b \approx 0$. Then we used the sample-mean complexity method (Section 4.5.5.4) to predict a. The values $a = 0.60$ and $b = 0$ were used.

Figure 4.2(c) shows $P(x)$ decaying exponentially with estimated K-complexity, though many low complexity outputs also have low probability. Prediction 1 agrees well with the data.

Prediction 2: We approximated that $b = 0$, as in Prediction 1. Then we used the sum-of-probabilities method (Section 4.5.5.1) to predict the value of a. The values $a = 0.59$ and $b = 0$ were used. The prediction also agrees well with the data.

The relation of probability to complexity that we observe does not appear to depend sensitively on how the inputs α_i are sampled. Figure 4.2(a) uses $\alpha_i \sim \mathcal{N}(0, 1)$ but analogous plots look qualitatively similar for different randomisations of α_i: Figure 4.5 in Section 4.10 depicts such analogous plots with $\alpha_i \sim U(-1, 1)$, some arbitrary less standard function $\alpha_i \sim \sin(U(-1, 1)) \cos(U(-1, 1))$, and finally $\alpha_i = \pm 1$ with equal probability.

Prediction 3: For Figure 4.5(c), additionally we show Prediction 3, where we make predictions *only* using the output complexities. That is, without making system specific assumptions, but rather only using the sum-of-probabilities method Section 4.5.5.1 and Eq. (4.43) to make a prediction directly based on output complexities (and not assuming or approximating $b = 0$). Such a prediction is possible in this case of the randomisation $\alpha_i = \pm 1$, because this implies there are relatively few inputs ($N_I = 2^{15} \sim 32000$), and so full enumeration of inputs cas easily be performed, which allows finding all the outputs. Having all the outputs means we can simply count them to find that $N_O = 662$, and then measure all complexities to find $\max(\tilde{K}(x))$. The values $a = 0.33$ and $b = 2.35$ were used in Prediction 3 (in contrast $a \approx 0.47$ and $b = 0$ were used for the other two quite accurate predictions in Figure 4.5(c))

Prediction 3 is not very accurate, but this may possibly be partly due to the fact the map is not strongly degenerate (as required by Eq. (4.21)). It may also be because

many outputs have probabilities far below the predicted upper bound.

Comment: Finding simplicity bias in the case of polynomials is especially striking because polynomials are so fundamental to mathematics (e.g. the Taylor series), hence supporting the very general nature of our findings.

4.6.2.4 Feed forward network

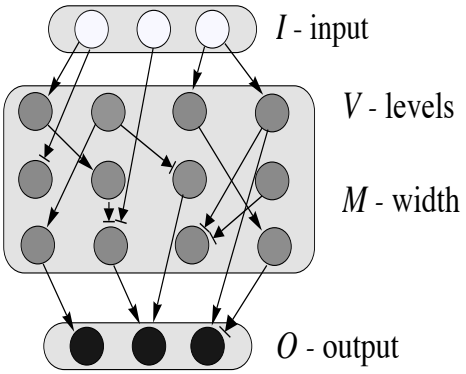
Model: This model [56] of information processing in a network has binary string stimuli (inputs) mapping to binary string responses (outputs), via a small feed-forward network (FFN). The model takes binary string inputs and processes them via a network of nodes, where nodes can regulate other nodes by positive, negative or no regulation (in practice, the model is realised by multiplying matrices and vectors). Using the notation of Ref. [56], the number of inputs is denoted by I, the number of outputs is denoted by O, the number of 'hidden levels' is denoted by H and the 'width' is denoted by M. See Figure 4.1 for an example diagram of the model.

Methods: To simplify the model slightly, we made two changes to the way we implemented the model as compared to the way it was implemented in Ref. [56]: Firstly, rather than sampling networks via a random walk of 'mutations' from a given starting network, we generated new random networks on each sample (additionally, our sampling approach is more appropriate for estimating $P(x)$ as compared to mutational random walk sampling). Secondly, we did not impose the constraint on the random networks that they should each have approximately the same average connectivity, as the original authors did. The parameters we used were: Number of inputs $I = 2^3 = 8$, length of outputs is 3, the number of hidden levels is $H = 1$ and width $M = 3$. We made 10^6 samples of the random networks.

Inspired by the application in Ref. [147], we used a Boolean function information content measure [27], which equals the entropy of the distribution over outputs, for a given choice of network. For example, if after fixing some network, and subsequently enumerating all inputs I, the outputs $(0,0,0)$, $(0,0,0)$, $(0,1,0)$ and $(1,0,1)$ were generated, then the probability distribution over the three different outputs would be 0.5, 0.25 and 0.25, which yields an entropy (hence complexity) of 1.5 bits.

Prediction: Figure 4.2(d) shows $P(x)$ decaying exponentially with estimated complexity, though many low complexity outputs also have low probability. We do not make a precise prediction of a and b in this example, as the complexity measure behaves quite differently as compared to K-complexity, e.g. this complexity measure drops to zero for 0^n or 1^n (instead of $\sim \log(n)$). Additionally, in this map it is not clear to what extent the mapping specifies the outputs, and it is not clear to what extent condition Eq. (4.19) would be satisfied in this map because it is not clear if we can view the map as a roughly constant/fixed rule-set. Finally, we did not gain a good estimate of N_O from sampling as N_I (i.e. the space of possible networks) is very large, which also hinders making predictions.

A related example to this FFN is Raman and Wagner's random logic gate map [147], which we outline in Section 4.10.



(a)

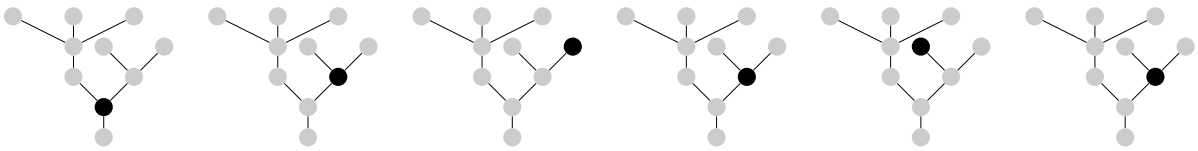
Figure 4.1: A cartoon illustrating the feed forward network (FFN) signalling model and how the input (white circles) is transmitted, through the V by M hidden units (grey circles), to the output (black circles). Arrows represent regulation.

4.6.2.5 L-Systems

Model: L-Systems [119] are a general modelling framework originally introduced for modelling plant growth, but now also used in computer graphics. Essentially, L-Systems consist of a string of different symbols which constitute production rules

for making some geometrical shape. We confined our investigation to non-cyclical graph (i.e. topological tree) outputs.

Methods: All valid L-Systems (of input length ≤ 9) which generate non-cyclical graphs were enumerated[4]. The outputs were coarse grained in two ways: For Figure 4.2(e), we used a method suggested by Ref. [124] to associate binary strings to any non-cyclical graph with a distinguished node called the 'root' (these graphs are known as *rooted trees*). Specifically, by walking along the branches, starting right, and recording whether it is going up (0) or down (1), a binary string representation of the tree is made. We illustrate the first few steps of generating a string for an example tree below



This method of assigning a binary string to a tree only takes the topology of the graph into account. As a small check that simplicity bias is not an artefact of our 'topology-only' representation of outputs, we also coarse grained the trees slightly less strongly, by recording the branching angle as a binary string: Choosing a discretisation of branching angles as $\pi/8$, there are only 16 possible angle values, and we can encode angles using numbers $0, \ldots, 16$ instead of simply $0, 1$. Because we want to preserve symmetry, we choose to use numbers from 0 to 8 and add two extra symbols $+, -$ indicating whether the angle is $\leq \pi$ or $> \pi$.

Both the topological and the more refined measure methods of coarse graining outputs generated binary strings, so to calculate output complexities we simply applied C_{LZ} to these strings.

[4]Note: Benjamin Frot performed the L-System computational analysis while undertaking a project under the supervision of K. Dingle. However, the suggestion for the binary representation (and hence complexity measure) for outputs was K. Dingle's. The tree figures were also drawn by B. Frot.

Prediction: Figure 4.2(e) with a more coarse and Figure 4.3(d) with a more fine grained measure show exponential decay in $P(x)$ with K-complexity, i.e. simplicity bias. Because we have the full list of outputs in both cases, we can calculate $\max(\tilde{K})$ and N_O directly, and hence the gradient a via Eq. (4.43). To estimate b, we used the calculated value of $P(x)$ for the highest probability (and simplest structure) output, and then used the method as in Section 4.5.5.3. The values $a = 0.13$ and $b = 2.41$ were used.

The upper bound prediction for Figure 4.2(e) is good, whereas for Figure 4.3(d), clear simplicity bias can be seen, but the prediction is less impressive as compared to the topology-only map. This less accurate prediction may be because in this map with more refined outputs, 82% of outputs only have *one* associated input, which is problematic in terms of seeing simplicity bias, as we gave as a condition that there should be $N_I \gg N_O$ (Eq. (4.21)). In contrast, for the 'topology only' representation, 62% of outputs had only one associated input. This is still high, but lower than 82%.

4.6.2.6 Simple ODE (GRN)

Model: This model [18] was designed to describe a simple genetic regulatory network (GRN), and has two output concentration curves, representing the concentration of mRNA and a protein. In our input-output framework, we take the inputs to be the values of the 6 parameters to this ODE system, and the output to be a single (discretised) curve depicting the concentration-time curve for the mRNA molecules.
Methods: The 6 parameters r_i, $i = 1, 2, ..., 6$ in the model were each allowed to assume the values $\{1/5, 2/5, 3/5, 4/5, 5/5\}$ (but the simplicity bias is not sensitive to the precise choice of values). Additionally, the four initial condition values in the model were chosen to each be equal to 1 or 2. Hence there were $N_I = 5^6 2^4 = 2.5 \times 10^5$ inputs. We performed complete enumeration of inputs, which allowed for obtaining exact values of N_O and $\max(\tilde{K})$. MATLAB was used to solve the differential equations. The curve is discretised in the same 'up-down' way as the polynomial curve above, but with $t \in [1, 30]$, $\delta t = 1$, and a resulting output string length of 30 bits.

Prediction: To make the prediction for $P(x)$ in Figure 4.2(f), firstly the gradient a was estimated via Eq. (4.43), using the values of N_O and $\max(\tilde{K})$ found from enumeration. To estimate b, we first measured the value of $P(x)$ for the simplest structure, and then using a, $P(x)$ and $\max(\tilde{K}(x))$ we used the method of Section 4.5.5.3. The values $a = 0.37$ and $b = 1.65$ were used.

Figure 4.2(f) shows the expected relation of probability decaying with increasing K-complexity. The prediction in this plot agrees very well with the data, though many outputs are far from their upper bound.

4.6.2.7 Circadian rhythm

Model: This model [180] describes a system of differential equations (of 6 equations) where the inputs are the values of the 15 parameters, and the output is a single curve depicting a concentration-time curve for a chemical (determining a circadian rhythm). **Methods:** The curve is discretised in the same ('up-down') way as the polynomial curve above. The original paper does not give ranges for the input parameters, so we chose the range of inputs to be the default values for the 15 parameters given in Ref. [180], but multiplied by one of $\{1/4, 2/4, 3/4, 4/4\}$, chosen with equal probability. Thus $N_I = 4^{15} \sim 10^9$. To generate Figure 4.2(g), partial sampling of 2×10^5 inputs were made. In this case, we chose $\delta t = 1$, $t \in [1, 25]$, which yields a length 25 bit string output. MATLAB was used to solve the differential equations, and for randomly sampling inputs.

As a small extension to the 'up-down' method, for this map we also tried discretising the concentration curves in what we call the 'up-down-flat' (UDF) method. In the UDF method, the curve is coarse grained to a trinary string made up of the alphabet {1, 0, -1}, representing a positive, zero and negative gradient. The string is generated as follows: For discrete values of $t = \delta t, 2\delta t, 3\delta t \ldots$, if $dy/dt > \gamma$ (with $\gamma > 0$) at time $j\delta t$, then a 1 is assigned to position j of the string; if $dy/dt < -\gamma$ at time $j\delta t$, then a -1 is assigned to position j of the string; otherwise a 0 is assigned. We choose a value of $\gamma \approx 0$, so that when $|dy/dt| < \gamma$ the curve can reasonably be called 'flat'. The precise choice of γ is somewhat arbitrary, and here we chose γ to be

10% of the mean absolute change in $|y((j+1)\delta t) - y(j\delta t)|$ over all samples and all j. In this case, we chose $\delta t = 1$, $t \in [0, 25]$, which yields a length 25 bit string output.

Prediction 1 and 2: Figure 4.2(g) shows the predicted relation of probability to estimated K-complexity. The predictions were done in the same way as for the polynomial curve (hence we keep the same nomenclature). The values $a = 0.53$ and $b = 0$ were used for Prediction 1. The values $a = 0.51$ and $b = 0$ were used for Prediction 2. Both prediction are very close to one another and compare favourably to the data.

Finally, we show in Figure 4.3(c) a plot of complexity vs. probability for the UDF method, which shows qualitatively the same simplicity bias behaviour as the model does when using the up-down method. We did not compare this method to any prediction.

4.6.2.8 Cell cycle

Model: Another example related to the two preceding is the model of Ref. [25], which describes part of the cell cycle for budding yeast. This famous model is made up of a large system of differential equations (~ 50 equations) where the inputs are the values of the > 130 parameters, and the outputs are curves depicting a concentration-time curve for different chemicals.

Methods: We used the MATLAB code available from

`http://mpf.biol.vt.edu/research/budding_yeast_model/pp/getwinpp_current_model.php`

to implement this model, in addition to using MATLAB for the sampling. As mentioned, the model describes a system of differential equations which determine various curves depicting concentration-time curves for biochemicals involved in cell cycle regulation. Of these biochemicals, we chose the concentration-time curve of Cdc6 as the output/phenotype, which was motivated only by the observation that the curve varies sufficiently to yield many different outputs of varying complexity. Further, we coarsely discretise the curves to binary strings following the 'up-down' method [190, 60] as above, yielding a binary string length of length 39.

In a model with parameters as inputs/genotypes, it is difficult to decide how to mutate or sample the parameters. This is because biologically realistic ranges for parameters are often not known, and additionally whether or not it is sensible to uniformly sample parameters from some range is also questionable, as the connection between random DNA sampling and random model-parameter sampling may well be highly non-linear and complex. Despite this, to make progress we must try *some* form of parameter sampling. Hence, inspired by the approach of the model authors (Chen *et al* [25]) in their robustness sampling for this model, we set all parameters to default values, and then sample by allowing each parameter to be scaled by the (random) factor $\sqrt{2}^{\zeta}$, where $\zeta \in \{-4, -3, -2, -1, 0, 1, 2, 3, 4\}$ with uniform probability (Chen *et al* allowed ζ to assume a larger range, but we limit the range to reduce the astronomical size of the genotype space). By sampling in this way for 2.5×10^5 parameter value genotypes, and discretising phenotypes as described, we generated the plot in Figure 4.2(h).

Prediction: We did not make a prediction for this model as we neither have have N_O nor $\max(\tilde{K}(x))$, and these are difficult to estimate from sampling due to the very large number of inputs and the (relatively) computationally demanding (hence time consuming) mapping, which involved numerically solving many differential equations.

Comment: Figure 4.2(h) shows a plot of probability vs. complexity for this model, which displays the same simplicity bias behaviour as the other models we have examined. The fact that simplicity bias appears in these three examples of differential equation systems, from very simple to very intricate models, supports the general applicability of our findings.

4.6.2.9 Bias, but not simplicity bias in a model for development

Model: Finally, we give an example of a map showing bias, but which is not simplicity bias: The Borenstein-Krakauer (BK) map [15] (described also in Section 2.2.4.3) consists of a binary vector input g (genotype) which is mapped in a many-to-one fashion to binary vector outputs p (phenotypes), via a matrix D representing a toy gene regulatory network. The square matrix D has entries either 1 or -1, which are

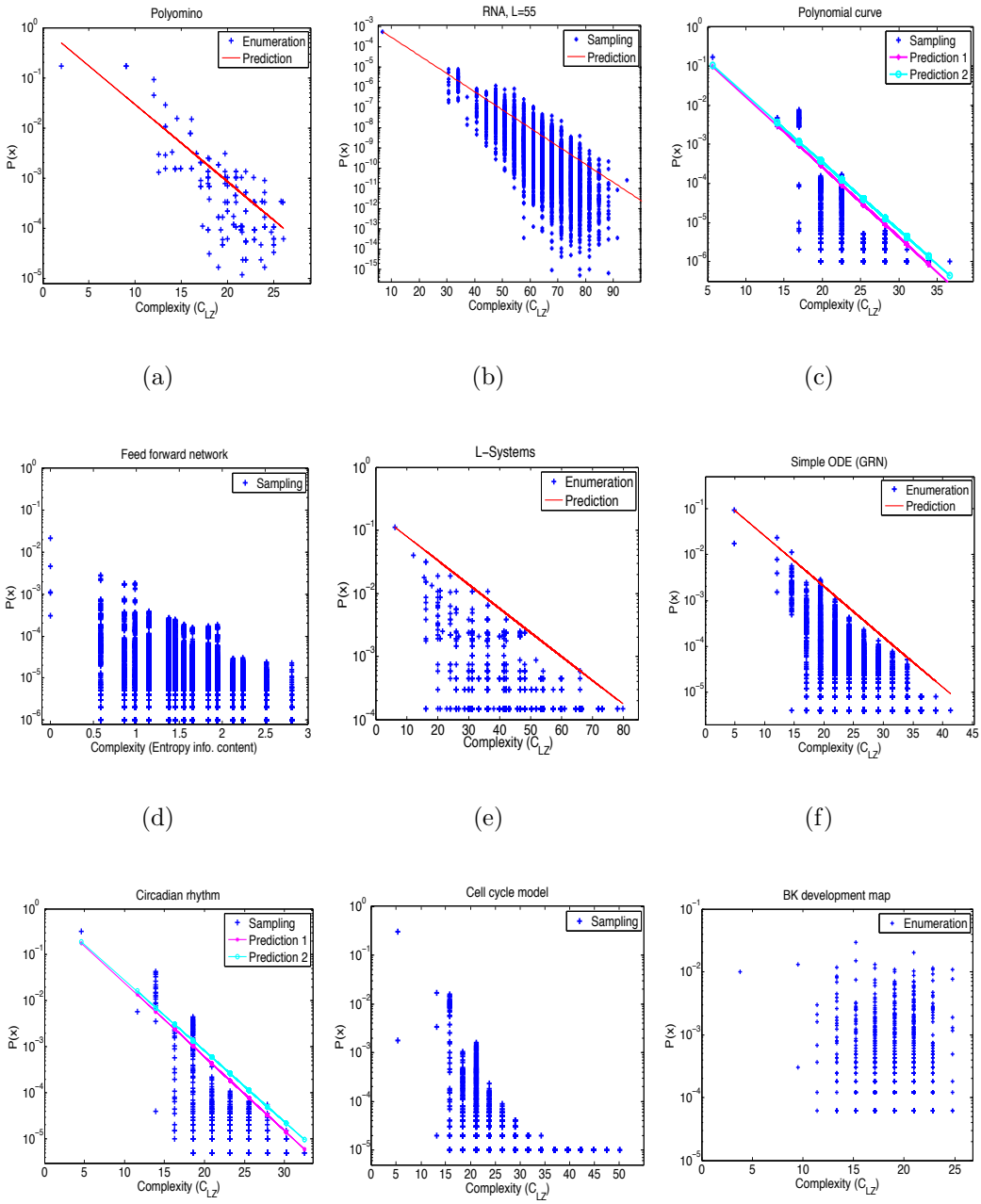


Figure 4.2: Decay in probability $P(x)$ with increasing K-complexity. See main text for explanation of models and their predictions. (a) A model of block self-assembly using polyominoes [90]; (b) RNA secondary structures of length $L = 55$ obtained from random genotype sampling; (c) Polynomial curve of degree 14, with randomly sampled coefficients; (d) Feed forward network (FFN) model of information/signal processing [56]; (e) L-Systems; (f) A simple ODE model of a GRN model [18]; (g) An ODE model of circadian rhythm [180]; (h) A detailed ODE model of a cell cycle [25]; and (i) Bias, but not simplicity bias, in the BK model of development [15].

chosen randomly. Output (phenotype) vectors are computed by multiplying g by the network interaction matrix, and applying the Heaviside function H to each row, i.e. $p = H(Dg)$.

Methods: We performed exhaustive enumeration of all $N_I = 2^{14} \approx 16000$ binary string inputs, and a random mapping matrix (D) was chosen (using MATLAB). In the realisation of Figure 4.2(i), $N_O = 992$. We calculated output complexities using C_{LZ}.

Prediction: This map produces strongly non-uniform output probabilities [15], but we find that the probabilities show little connection to estimated output complexities, see Figure 4.2(h).

This lack of correlation can be rationalised in two ways: Intuitively, we can see that the pattern of 1s and 0s in p is strongly determined by the particular choice of mapping matrix D. So, for example having many -1 entries in row i of $\mathcal{D}$ implies that entry i of vector p will be 0 with high probability. Such sensitive dependence on the mapping was explicitly highlighted above (Section 4.3.3.1) as a problem for observing simplicity bias. From another perspective, we can see that the condition of Eq. (4.19) will *not* be satisfied in this case: For a typical map, specifying D yields $K(\mathcal{A}) = O(l(g)^2)$, while any output will have $K(x) \leq O(l(g))$. Hence asymptotically $K(\mathcal{A}) \gg K(x)$, and so the BK model does not constitute a simple map, and so this model is not a good candidate for showing simplicity bias.

Comment: It is gratifying that the one map for which our criterion for a 'simple' map clearly does not hold shows bias, but not simplicity bias. This result helps confirm our earlier analysis, and it would be interesting to study models like this to see what other ways bias can emerge.

4.7 Why abstract and asymptotic results from AIT may still apply to concrete maps

In Section 4.1 we mentioned that there are some theoretical difficulties for applied AIT, but (temporarily ignoring these difficulties) we would take an experimental

approach, and just try to apply the theory to a range of maps to see how well it performs. It seems clear from the previous section that the results of AIT apply surprisingly well. In that light, we return to look at these difficulties in more detail, and suggest why in fact they may not be such great obstacles for applied AIT.

4.7.1 UTMs

Given that AIT results usually assume the presence of UTMs, and many real-world mappings like $\mathcal{A}$ are not UTMs, one may question the applicability of AIT results for these real-world maps. Nevertheless, we showed that properties from UTMs and AIT, in the asymptotic limit, give useful and sometimes quite accurate predictions for the non-UTMs maps just examined. Now, deriving rigorous conditions under which these useful and predictions hold is difficult, and will hopefully become an active subject of future research. Nonetheless, in the meanwhile, we offer some preliminary thoughts on this topic below.

4.7.2 Incomputability

A frequently stated difficulty for applied AIT is the fact that K-complexity is incomputable [117]. Recall that for $K(x)$ to be incomputable means that in general there cannot exist any method that takes x and computes its K-complexity. However, in a number of real-world settings, K-complexity has been successfully approximated, in the sense that approximations behave in a manner expected of the true K-complexity. Example settings include: DNA and phylogeny studies [152, 59, 116], plagiarism detection [26], clustering music [32], and financial market analysis [197]; see also Vitányi [181] for a recent review. These successes (in addition to our work) suggest that K-complexity can be usefully approximated, even if the exact value of K-complexity is incomputable.

A natural question then is to ask why these approximations have been successful, which again we hope is addressed in future research. In the meantime, one perspective is that for $K(x)$ to be termed 'incomputable' is a statement about an inability to *know* (by finite algorithmic means) the value of $K(x)$, not an inability to estimate

it. Expanding on this, for a given string x, there is no theoretical reason why an estimation of $K(x)$ (using, say, a compression algorithm) must be incorrect, only that one cannot *prove* and hence *know* that it is the correct value. Given this, it is interesting to ask how good such estimates of $K(x)$ would be. This is an important topic which we will not explore here, but one significant result to highlight is that with high probability [35]

$$K(x) \approx nS(k/n) + O(\log(n)) \tag{4.57}$$

for uniformly sampled $x \in \{0,1\}^n$, where $S(k/n)$ is the Shannon entropy (in bits) of a length n bit string with k 1s. Moreover, such complexity estimates as we use based on the Lempel-Ziv measure [109] or other compression algorithms can be expected to yield close estimates to $K(x)$ with even higher probability (as they are better at detecting pattern in strings). There will still remain low K-complexity strings which have high entropy/complexity (such as π), but by Eq. (4.57), these must be relatively rare. So, by using common (computable) complexity measures, it appears that good estimates to $K(x)$ can be expected, for typical strings, even if the quantity is incomputable.

4.7.3 Asymptotic results

AIT is a body of asymptotically valid results, that is, results hold only up to $O(1)$ constant terms or logarithmic terms. For example, the K-complexity of a string is only UTM independent up to $O(1)$ terms. As such, AIT results can be guaranteed only in the case of large/long structures with high K-complexity values, where constant or logarithmic terms are negligible [35, 117].

This asymptotic condition presents a difficulty for applied AIT, as for any given real-world application, it may not be obvious whether the structure in question is 'large' enough to have confidence in the predictions of the relevant AIT-derived results. Having said this, we argue below from different perspectives that the presence of $O(1)$ terms should not strongly detract from the applicability of AIT and our results for maps.

Firstly, despite the potential for $O(1)$ terms to strongly influence K-complexity estimates for short bit strings, recent numerical experiments applying AIT to very short strings of $\sim$10 bits [41, 198, 40] demonstrated that results claimed by AIT for 'large' objects can also be observed for small objects (cf. also Ref. [199] for similar work for graphs). These findings, along with our work and the other successful applications of AIT mentioned above, suggest that in many settings the presence of $O(1)$ terms does not preclude the possibility of making accurate predictions. That is, operationally one may be able to ignore the $O(1)$ terms more often than one might think at first.

Secondly, the AIT results we highlight concern fundamental aspects of maps. For strong effects such as coding theorem behavior with exponential variation in $P(x)$, it is reasonable to assume that behaviour that is qualitatively similar will also be observed in small systems. It may be that more subtle predictions of AIT are more sensitive to these $O(1)$ terms.

Thirdly, because often the same rule-set is used for mapping both large and small systems, then if the theory predicts simplicity bias for large systems, then it is quite reasonable — given the same mapping rules — to predict the simplicity bias also in the small systems. For example, if coding theorem behaviour is predicted and observed for the RNA sequence to structure map with L very large, then as the same rule-set defines the map for all sizes of L, it is reasonable to predict coding theorem-like behaviour for 'small' L, provided perhaps that L is not so small that idiosyncrasies of the mapping itself begin to dominate.

Fourthly, it is worth stressing that AIT does not predict that relations involving O(1) terms will *not* hold for small systems, but rather that their presence suggests that one can only guarantee their negligibility for large systems.

Nevertheless, the arguments listed above remain very preliminary and would need to be worked out in much more detail in future work.

4.7.4 Intuitive connection of probability and complexity

We have used rather abstract results from AIT to make a connection between probability and structural complexity in simple maps. While these results have facilitated a quantitative connection, in fact the general inverse relation between complexity and probability can also be argued intuitively (see also Refs. [90, 89]). We illustrate now with two examples.

Firstly, it can be seen intuitively that if, say, a polyomino shape is modular, rather than encoding each part of the structure separately, the GP map allows a more efficient encoding of the shape by coding only for a few blocks which then are repeatedly used to construct the entire shape. Less constraints on the genotype equates to more genotypes being compatible with such a phenotype, hence a higher probability of generating the phenotype on choosing a random genotype. Similarly small shapes — which are intuitively simpler than larger shapes — can be expected to require less coding from the genotype by dint of having less specifiable structure, and hence typically have a high probability.

Secondly, if we consider a 'complex' RNA secondary structure (SS), then intuitively such a structure would have many motifs, including many stacks and bulges. We saw in the previous chapter that log NSS (hence probability) correlates quite strongly (and linearly) with the number of stacks in an RNA SS. Therefore we can expect 'complex' RNA to also have exponentially smaller probabilities than 'simpler' RNA with only one or a few stacks.

* * *

In summary, despite these potential difficulties for applied AIT, we see that in each case the difficulty may be at least partially overcome. Additionally, we pointed out that some of the theoretical results of AIT are intuitively reasonable, suggesting that the essence of the theory should apply more generally than only in abstract UTM settings. Hence, we speculate that some 'reasonable' mappings like $\mathcal{A}$ which are not UTMs, nonetheless may still share some behaviours of UTMs, even if only

approximately. Clearly there is much work to be done, however, in establishing or refuting these conjectures. We nevertheless present them as potentially fruitful.

4.8 Discussion

4.8.1 Summary

We have shown via a novel application of AIT that *a priori* estimates of the probability $P(x)$ of a given output x in a large class of 'simple' input-output maps can be made, or at least $P(x)$ can be bounded tightly. Additionally, this work allows the prediction of the total number of outputs (N_O) from estimating the gradient of a complexity-probability plot and the maximum complexity of outputs. To our knowledge, little if any theory existed or was widely known for such predictions, previous to this work. Hence our work may open up many new research avenues. While the predictions that we make may not be extremely exact, we stress that given the minimal system specific assumptions we make, the success of the predictions is striking. Further, this work marks a step forward in AIT applied to the sciences, as quantitative — and sometimes precise — predictions have been made, rather than only qualitative statements or trends. Also, we do not use system-specific complexity measures (which assume knowledge of the mapping process), rather we work directly from output complexities, as measured by generic compression-based complexity measures.

4.8.2 Open questions and future work

Many research directions could be undertaken to advance and apply the results of this Chapter. However, some specific future questions and directions are:
i) At present we do not have a straightforward practical checklist which predicts when the conditions for simplicity bias will and will not be met for a given map A (though we can make firm predictions sometimes). Hence this should be investigated in future.
ii) Several of the examples used the same 'up-down' method of discretising the output curves which is rather coarse; further work should explore other less coarse complexity measures for curves.

iii) In some maps that we examined, the output probabilities appears to follow the AIT upper bound with fidelity, while many of the other maps showed many outputs with probabilities far below their upper bounds. It would be interesting to get a better understanding of this phenomenon, and to be able to predict when either case occurs.

iv) In this Chapter we only made predictions for simple maps where $K(x|\mathcal{A}) \sim K(x)$. A natural extension would be to try to also make predictions for maps in which $K(x|\mathcal{A}) \nsim K(x)$, and also to investigate ways to estimate $K(x|\mathcal{A})$ accurately (as opposed to estimating just $K(x)$ by compression).

4.9 Appendix: A lower bound on $P(x)$ for computable maps

We give a probabilistic lower bound on Eq. (4.17), which essentially states that for most inputs, the corresponding output's probability $P(x)$ will be close to its upper bound (Eq. (4.17)). We show this by following the method of Ref. [117], which uses Markov's inequality to derive a (weak) lower bound on $P(x)$, as follows: Let the function

$$f(x) = \frac{2^{-K(x|\mathcal{A})+O(1)}}{P(x)} \tag{4.58}$$

Hence the $P-$expected value of $f(x)$, $\mathcal{E}$, is

$$\mathcal{E} = \sum_{i=1}^{N_O} P(x_i) f(x_i) \tag{4.59}$$

$$= \sum_{i=1}^{N_O} 2^{-K(x_i|\mathcal{A})+O(1)} \tag{4.60}$$

Clearly $\mathcal{E}$ is finite and $f(x) \geq 0$, so Markov's inequality implies

$$\sum_{i=1}^{N_O} \{P(x_i) : f(x_i) > \mathcal{E}r\} < \frac{1}{r} \tag{4.61}$$

for $r > 0$, which implies that

$$\sum_{i=1}^{N_O} \left\{ P(x_i) : \frac{2^{-K(x_i|\mathcal{A})+O(1)}}{\mathcal{E}r} \leq P(x_i) \right\} \geq 1 - \frac{1}{r} \tag{4.62}$$

and so with P-probability (i.e. sampling an input) at least $1 - \frac{1}{r}$, we have

$$\frac{2^{-K(x|\mathcal{A})+O(1)}}{\mathcal{E}r} \leq P(x) \tag{4.63}$$

for $x \in O$. Taking logarithms of Eqs. (4.63) and (4.17) yields,

$$\begin{aligned} K(x|\mathcal{A}) + O(1) &\leq \log(1/P(x)) \leq K(x|\mathcal{A}) + \log(\mathcal{E}r) + O(1) && (4.64) \\ \Rightarrow O(1) &\leq \log(1/P(x)) - K(x|\mathcal{A}) \leq \log(r) + \log(\mathcal{E}) + O(1) && (4.65) \end{aligned}$$

and so

$$\sum_{i=1}^{N_O} \{P(x_i) : \log(1/P(x_i)) - K(x_i|\mathcal{A}) > \log(r) + \log(\mathcal{E}) + O(1)\} < \frac{1}{r} \tag{4.66}$$

This result is only constraining if $\mathcal{E}$ is not very large. In fact we know that $\mathcal{E}$ asymptotically small since

$$\begin{aligned} \mathcal{E} &= \sum_{i=1}^{N_O} 2^{-K(x_i|\mathcal{A})+O(1)} && (4.67) \\ &< \sum_{x\in\{0,1\}^*} 2^{-K(x|\mathcal{A})+O(1)} && (4.68) \\ &= O(1) && (4.69) \end{aligned}$$

by Kraft's inequality [14], where $\{0,1\}^*$ is the set of all possible binary strings. So with high probability — that is, for nearly all inputs — $\log(1/P(x))$ is close to $K(x|\mathcal{A})$. However this does not necessarily mean that $\log(1/P(x))$ is close to $K(x|\mathcal{A})$ for most *outputs*.

4.10 Appendix: Further examples and figures

Here we give a few extra examples of simplicity bias.

Figure 4.3(a) shows a plot of predicted probabilities for the polyomino model as described in Section 4.6.2.1, but for a different choice of parameters, $S_{2,8}$. The prediction is reasonable, but not as striking as in the main text, though it is difficult to say how well the $P(x)$ prediction compares to the data with so few data points.

Figure 4.3(b) shows a logic gate input-output process. The probabilities and complexities were taken from the supplementary material to Raman and Wagner's

paper [147]. They did not plot the data in the manner we have done, nor was any simplicity bias-style connection between probability and complexity claimed. It is important to stress that these data points in the figure were chosen by Raman and Wagner to display a range of complexities. As such, this plot must be taken 'with a pinch of salt', because the fact that they were hand-picked may influence the correlation. The complexity here is measured as Boolean function information content [27], which is related to the number of different logic functions the circuit can execute (see the FFN example map in main text for more detail on this measure). Hence, this is a different complexity measure to the one based on LZ compression, which we have used extensively elsewhere in this chapter.

Figure 4.3(c) shows probability versus complexity for a model of a circadian rhythm, see Section 4.6.2.7. Here, instead of using the 'up-down' discretisation we have used the up-down-flat (UDF) discretisation method, as described in the main text. We again see simplicity bias, but do not attempt any direct predictions of $P(x)$.

Figure 4.3(d) shows a plot of L-Systems for less coarse grained outputs, as compared to the topology-only method as described in the main text, Section 4.6.2.5.

Figure 4.5 shows data for polynomial curves using different randomisation methods for the inputs; see Section 4.6.2.3 for details. Despite the varying randomisations, qualitatively similar relationships of complexity to probability can be observed.

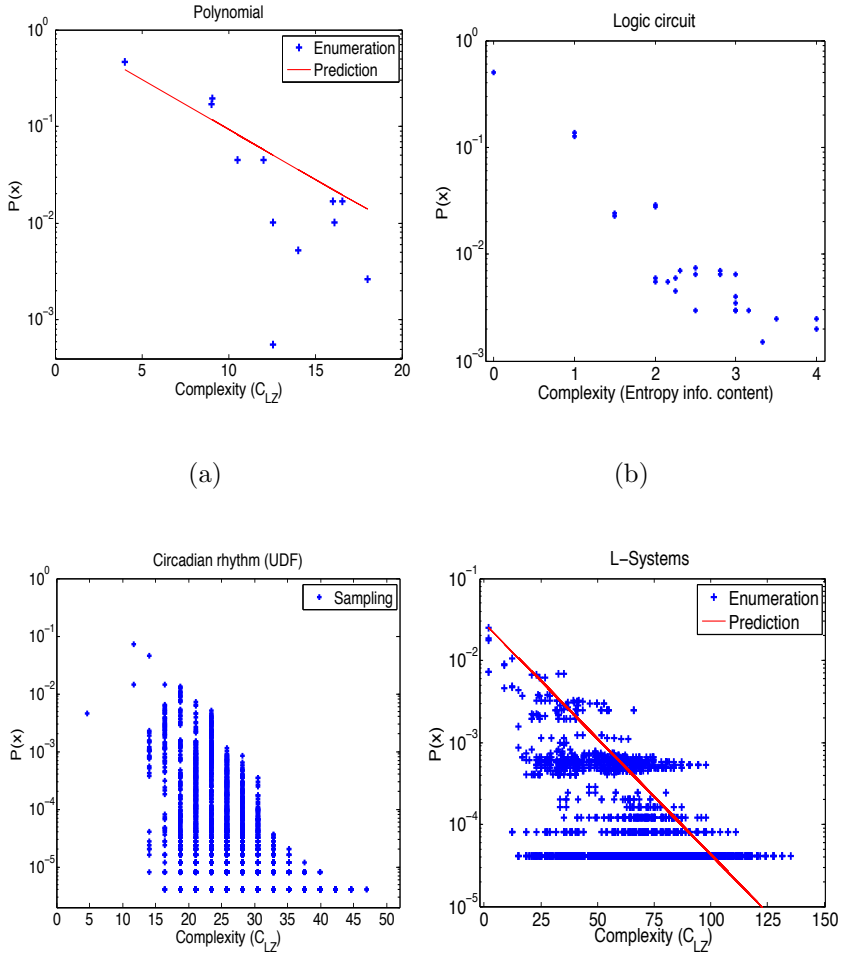
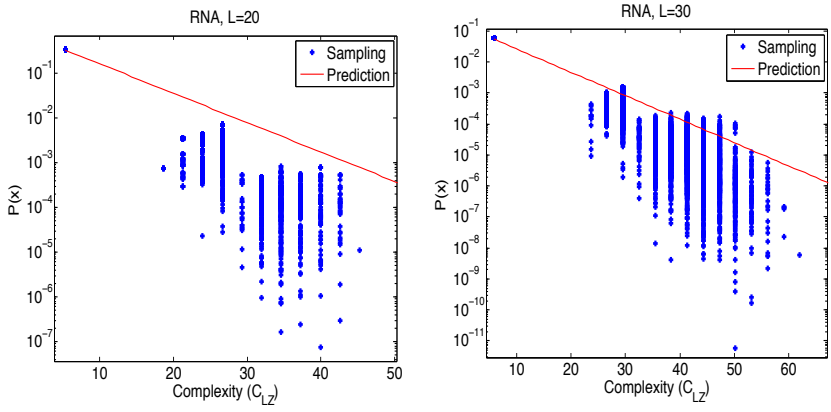
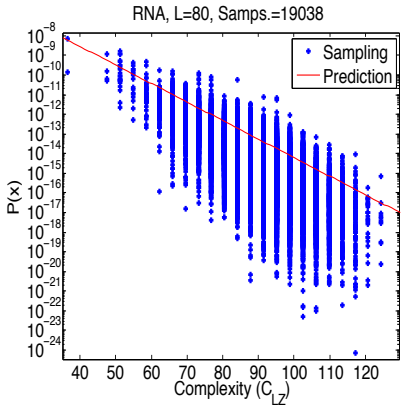


Figure 4.3: Decay in probability $P(x)$ with increasing K-complexity. (a) Polyominoes, $S_{2,8}$; (b) Logic gate; (c) Circadian rhythm, using the up-down-flat (UDF) method; (d) L-Systems, with less coarse grained outputs.

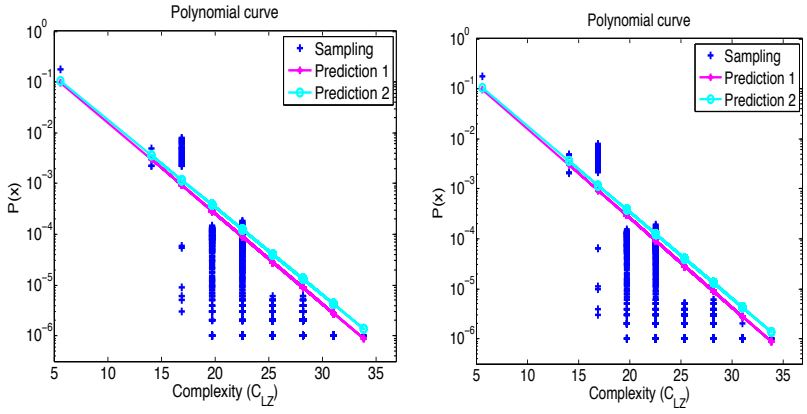


(a) (b)



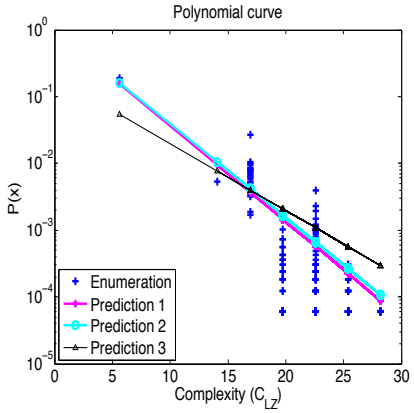
(c)

Figure 4.4: Decay in probability $P(x)$ with increasing K-complexity in RNA. (a) $L = 20$ ($\sim$5000 samples); (b) $L = 30$ ($\sim$5000 samples); (c) $L = 80$ ($\sim$20000 samples). Coding theorem-like behaviour becomes more pronounced for longer RNA.



(a)

(b)



(c)

Figure 4.5: Decay in probability $P(x)$ with increasing K-complexity in polynomial curves. See main text for predictions. (a) $\alpha_i \sim U(-1,1)$; (b) $\alpha_i \sim \sin(U(-1,1)\cos(U(-1,1)$; (c) $\alpha_i = 1, -1$ with equal probability.

Chapter 5

Conclusion

We showed that many different biological genotype-phenotype maps – ranging from the genetic code to molecular systems like RNA secondary structure to larger scale systems like gene regulatory networks or even neural systems – exhibit phenotypic bias. That is, certain phenotypes have many more genotypes mapping to them than others, i.e. they have larger neutral set sizes (NSS) than others. Motivated by the breadth and diversity of examples which we studied, we hypothesised that bias is a common property of genotype-phenotype maps (of the form considered). We also reviewed many studies which point to the significant role of NSS in influencing evolutionary dyamics and outcomes, even possibly overriding selection pressures from fitness advantages. That is, in some cases less fit phenotypes with larger NSS can outcompete 'fitter' phenotypes with smaller NSS, and come to dominate a population.

The importance of NSS in relation to evolutionary outcomes was explored in detail for RNA secondary structures. Specifically, we proposed an analytic form for the full distribution of NSS, showed that random RNA sequences almost always fold to secondary structures within a narrow range of NSS values, and found a striking similarity between natural and random RNA in terms of their NSS. This suggests that the evolution of natural RNA structures is strongly constrained, with NSS confining the evolutionary 'search' to a small range of possible structures.

In view of the above, we suggest that phenotypic bias should be incorporated more fully into the theory and models pertaining to evolutionary dynamics.

Finally, we proposed a method for estimating a phenotype's NSS which can be used in cases where deriving NSS by computational sampling or mathematical analysis of a given genotype-phenotype map is hard or impossible. The method was based on a relatively esoterical field known as algorithmic information theory (AIT). While our estimates were not always precise, and really formed a bound on a phenotype's NSS, the predictions were sometimes surprisingly good. Further, although this thesis was not intended as a study of applied AIT, the success of our (and others') application of AIT to real-world systems is perhaps more interesting and significant than the specific application to GP maps that we have used AIT for. Indeed, these successes of applied AIT should motivate future research for application of AIT into other areas of mathematics, science, and engineering.

Bibliography

[1] S.E. Ahnert, I.G. Johnston, T.M.A. Fink, J.P.K. Doye, and A.A. Louis. Self-assembly, modularity, and physical complexity. *Physical Review E*, 82(2):026117, 2010.

[2] P. Alberch. Ontogenesis and morphological diversification. *American Zoologist*, 20(4):653–667, 1980.

[3] P. Alberch. From genes to phenotype: dynamical systems and evolvability. *Genetica*, 84(1):5–11, 1991.

[4] R. Albert and H.G. Othmer. The topology of the regulatory interactions predicts the expression pattern of the segment polarity genes in drosophila melanogaster. *Journal of Theoretical Biology*, 223(1):1–18, 2003.

[5] U. Alon. *An introduction to systems biology: design principles of biological circuits*, CRC press, 2007.

[6] D.D. Axe. Estimating the prevalence of protein sequences adopting functional enzyme folds. *Journal of molecular biology*, 341(5):1295–1315, 2004.

[7] M.H. Bailor, X. Sun, and H.M. Al-Hashimi. Topology links RNA secondary structure with global conformation, dynamics, and adaptation. *Science*, 327(5962):202, 2010.

[8] M.H. Bailor, A.M. Mustoe, C.L. Brooks III, and H.M. Al-Hashimi. Topological constraints: using RNA secondary structure to model 3D conformation, folding

pathways, and dynamic adaptation. *Current Opinion in Structural Biology*, 21(3):296 – 305, 2011.

[9] A. Bairoch. The enzyme database in 2000. *Nucleic acids research*, 28(1):304–305, 2000.

[10] B.G. Barrell, A.T. Bankier, and J. Drouin. A different genetic code in human mitochondria. *Nature*, 282(5735):189, 1979.

[11] U. Bastolla, M. Porto, M.H. Eduardo Roman, and M.H. Vendruscolo. Connectivity of neutral networks, overdispersion, and structural conservation in protein evolution. *Journal of molecular evolution*, 56(3):243–254, 2003.

[12] U. Bastolla, H.E. Roman, and M. Vendruscolo. Neutral evolution of model proteins: diffusion in sequence space and overdispersion. *Journal of theoretical biology*, 200(1):49, 1999.

[13] P. Beldade, K. Koops, and P.M. Brakefield. Developmental constraints versus flexibility in morphological evolution. *Nature*, 416(6883):844–847, 2002.

[14] C.H. Bennett, P. Gács, M. Li, P.M.B. Vitányi, and W.H. Zurek. Information distance. *Information Theory, IEEE Transactions on*, 44(4):1407–1423, 1998.

[15] E. Borenstein and D.C. Krakauer. An end to endless forms: epistasis, phenotype distribution bias, and nonuniform evolution. *PLoS Comput Biol*, 4(10):e1000202, 2008.

[16] E. Bornberg-Bauer. How are model protein structures distributed in sequence space? *Biophysical journal*, 73(5):2393–2403, 1997.

[17] E. Bornberg-Bauer and H.S. Chan. Modeling evolutionary landscapes: mutational stability, topology, and superfunnels in sequence space. *Proceedings of the National Academy of Sciences*, 96(19):10689, 1999.

[18] K.S. Brown and J.P. Sethna. Statistical mechanical approaches to models with many poorly known parameters. *Physical review E*, 68(2):021904, 2003.

[19] C.S. Calude. *Information and randomness: An algorithmic perspective.* Springer, 2002.

[20] N.A. Campbell and J.B. Reece. *Biology. 8th Ed.* San Francisco, Pearson Education, 2008.

[21] J.M. Carothers, S.C. Oestreich, J.H. Davis, and J.W. Szostak. Informational complexity and functional activity of RNA structures. *Journal of the American Chemical Society*, 126(16):5130–5137, 2004.

[22] S.B. Carroll. Evolution at two levels: on genes and form. *PLoS Biology*, 3(7):1159, 2005.

[23] G.J. Chaitin. *Algorithmic information theory.* Cambridge University Press, 1987.

[24] G.J. Chaitin. A theory of program size formally identical to information theory. *Journal of the ACM (JACM)*, 22(3):329–340, 1975.

[25] K.C. Chen, L. Calzone, A. Csikasz-Nagy, F.R. Cross, B. Novak, and J.J. Tyson. Integrative analysis of cell cycle control in budding yeast. *Molecular Biology of the Cell*, 15(8):3841, 2004.

[26] X. Chen, B. Francia, M. Li, B. Mckinnon, and A. Seker. Shared information and program plagiarism detection. *Information Theory, IEEE Transactions on*, 50(7):1545–1551, 2004.

[27] K.T. Cheng and V.D. Agrawal. An entropy measure for the complexity of multi-output boolean functions. In *Proceedings of the 27th ACM/IEEE Design Automation Conference*, pages 302–305. ACM, 1991.

[28] C. Chiarabelli, J.W. Vrijbloed, D. De Lucrezia, R.M. Thomas, P. Stano, F. Polticelli, T. Ottone, E. Papa, and P.L. Luisi. Investigation of de novo totally random biosequences, part ii. *Chemistry & biodiversity*, 3(8):840–859, 2006.

[29] C. Chothia and A.M. Lesk. The relation between the divergence of sequence and structure in proteins. *The EMBO journal*, 5(4):823, 1986.

[30] T. Chouard. Beneath the surface. *Nature*, 456:300–303, 2008.

[31] F.B. Churchill. William Johannsen and the genotype concept. *Journal of the History of Biology*, 7(1):5–30, 1974.

[32] R. Cilibrasi, P. Vitányi, and R. Wolf. Algorithmic clustering of music based on string compression. *Computer Music Journal*, 28(4):49–67, 2004.

[33] I. Coluzza. A coarse-grained approach to protein design: learning from design to understand folding. *PloS one*, 6(7):e20853, 2011.

[34] I. Coluzza, J.T. MacDonald, M.I. Sadowski, W.R. Taylor, and R.A. Goldstein. Analytic markovian rates for generalized protein structure evolution. *PloS one*, 7(5):e34228, 2012.

[35] T.M. Cover and J.A. Thomas. *Elements of information theory*. John Wiley and Sons, 2006.

[36] M.C. Cowperthwaite, E.P. Economo, W.R. Harcombe, E.L. Miller, and L.A. Meyers. The ascent of the abundant: how mutational networks constrain evolution. *PLoS computational biology*, 4(7):e1000110, 2008.

[37] F.H.C. Crick. The origin of the genetic code. *Journal of molecular biology*, 38(3):367–379, 1968.

[38] E.H. Davidson. Emerging properties of animal gene regulatory networks. *Nature*, 468(7326):911–920, 2010.

[39] R. Dawkins. *The Blind Watchmaker*. Penguin, 2006.

[40] J.-P. Delahaye and H. Zenil. Towards a stable definition of Kolmogorov-Chaitin complexity. *arXiv preprint arXiv:0804.3459*, 2008.

[41] J.-P. Delahaye and H. Zenil. Numerical evaluation of algorithmic complexity for short strings: A glance into the innermost structure of algorithmic randomness. *Appl. Math. Comput.*, 219:63–77, 2012.

[42] M. Di Giulio. The origin of the genetic code: theories and their relationships, a review. *BioSystems*, 80(2):175, 2005.

[43] K.A. Dill. Polymer principles and protein folding. *Protein Science*, 8(6):1166–1180, 1999.

[44] K.A. Dill. Dominant forces in protein folding. *Biochemistry*, 29(31):7133–7155, 1990. PMID: 2207096.

[45] T. Dobzhansky and O. Pavlovsky. An experimental study of interaction between genetic drift and natural selection. *Evolution*, pages 311–319, 1957.

[46] J. Dolezel, J. Bartos, H. Voglmayr, J. Greilhuber. Nuclear DNA content and genome size of trout and human. *Cytometry. Part A: the journal of the International Society for Analytical Cytology*, 51(2):127, 2003.

[47] J.A. Draghi, T.L. Parsons, G.P. Wagner, and J.B. Plotkin. Mutational robustness can facilitate adaptation. *Nature*, 463(7279):353–355, 2010.

[48] D.T.F. Dryden, A.R. Thomson, and J.H. White. How much of protein sequence space has been explored by life on earth? *Journal of The Royal Society Interface*, 5(25):953–956, 2008.

[49] G.M. Edelman and J.A. Gally. Degeneracy and complexity in biological systems. *Proceedings of the National Academy of Sciences*, 98(24):13763, 2001.

[50] D. Elliott and M. Ladomery. *Molecular biology of RNA*. Oxford University Press New York, 2011.

[51] J.L. England, B.E. Shakhnovich, and E.I. Shakhnovich. Natural selection of more designable folds: a mechanism for thermophilic adaptation. *Proceedings of the National Academy of Sciences*, 100(15):8727, 2003.

[52] J.L. England and E.I. Shakhnovich. Structural determinant of protein designability. *Physical review letters*, 90(21):218101, 2003.

[53] C. Espinosa-Soto, P. Padilla-Longoria, and E.R. Alvarez-Buylla. A gene regulatory network model for cell-fate determination during arabidopsis thaliana flower development that is robust and recovers experimental gene expression profiles. *The Plant Cell Online*, 16(11):2923–2939, 2004.

[54] P. Fara. *Science: a four thousand year history*. Oxford University Press, 2009.

[55] A. Fernández and M. Lynch. Non-adaptive origins of interactome complexity. *Nature*, 474(7352):502–505, 2011.

[56] P. Fernández and R.V. Solé. Neutral fitness landscapes in signalling networks. *Journal of The Royal Society Interface*, 4(12):41, 2007.

[57] E. Ferrada and A. Wagner. Protein robustness promotes evolutionary innovations on large evolutionary time-scales. *Proceedings of the Royal Society B: Biological Sciences*, 275(1643):1595–1602, 2008.

[58] E. Ferrada and A. Wagner. Evolutionary innovations and the organization of protein functions in genotype space. *PloS one*, 5(11):e14172, 2010.

[59] P. Ferragina, R. Giancarlo, V. Greco, G. Manzini, and G. Valiente. Compression-based classification of biological sequences and structures via the universal similarity metric: experimental assessment. *BMC bioinformatics*, 8(1):252, 2007.

[60] T.M.A. Fink, K. Willbrand, and F.C.S. Brown. 1-D random landscapes and non-random data series. *EPL (Europhysics Letters)*, 79(3):38006, 2007.

[61] R.A. Fisher. *The genetical theory of natural selection*. Clarendon, 1930.

[62] W. Fontana. Modelling 'evo-devo' with RNA. *BioEssays*, 24(12):1164–1177, 2002.

[63] W. Fontana, D.A.M. Konings, P.F. Stadler, and P. Schuster. Statistics of RNA secondary structures. *Biopolymers*, 33(9):1389–1404, 1993.

[64] A. Fontdevila. *The Dynamic Genome: A Darwinian Approach*. Oxford University Press, Oxford, 2011.

[65] S.J. Freeland and L.D. Hurst. The genetic code is one in a million. *Journal of molecular evolution*, 47(3):238–248, 1998.

[66] S.J. Freeland, R.D. Knight, L.F. Landweber, and L.D. Hurst. Early fixation of an optimal genetic code. *Molecular Biology and Evolution*, 17(4):511–518, 2000.

[67] P. Gács. *Lecture notes on descriptional complexity and randomness*. Boston University, Graduate School of Arts and Sciences, Computer Science Department, 1988.

[68] V.E. Galkin, X. Yu, J. Bielnicki, J. Heuser, C.P. Ewing, P. Guerry, and E.H. Egelman. Divergence of quaternary structures among bacterial flagellar filaments. *Science*, 320(5874):382, 2008.

[69] D. Gilis, S. Massar, N.J. Cerf, and M. Rooman. Optimality of the genetic code with respect to protein stability and amino-acid frequencies. *Genome Biology*, 2(11):49–1, 2001.

[70] S.J. Gould and R.C. Lewontin. The spandrels of San Marco and the Panglossian paradigm: a critique of the adaptationist programme. *Proceedings of the Royal Society of London. Series B, Biological Sciences*, 205(1161):581–598, 1979.

[71] S.J. Gould. *The structure of evolutionary theory*. Belknap Press, 2002.

[72] S.F. Greenbury, I.G. Johnston, A.A. Louis, and S.E. Ahnert. A tractable genotype–phenotype map modelling the self-assembly of protein quaternary structure. *Journal of The Royal Society Interface*, 11(95):20140249, 2014.

[73] L.H. Greene, T.E. Lewis, S. Addou, A. Cuff, T. Dallman, M. Dibley, O. Redfern, F. Pearl, R. Nambudiry, A. Reid, et al. The cath domain structure database: new protocols and classification levels give a more comprehensive resource for exploring evolution. *Nucleic acids research*, 35(suppl 1):D291–D297, 2007.

[74] W. Gruner, R. Giegerich, D. Strothmann, C. Reidys, J. Weber, I.L. Hofacker, P.F. Stadler, and P. Schuster. Analysis of RNA sequence structure maps by exhaustive enumeration ii. structures of neutral networks and shape space covering. *Monatshefte fur Chemie/Chemical Monthly*, 127(4):375–389, 1996.

[75] X. Gu, D. Hewett-Emmett, and W.H. Li. Directional mutational pressure affects the amino acid composition and hydrophobicity of proteins in bacteria. *Genetica*, 102:383–391, 1998.

[76] H.H. Guo, J. Choe, and L.A. Loeb. Protein tolerance to random amino acid change. *Proceedings of the National Academy of Sciences*, 101(25):9205–9210, 2004.

[77] M.W. Hahn. Toward a selection theory of molecular evolution. *Evolution*, 62(2):255–265, 2008.

[78] J.B.S. Haldane. *The causes of evolution*. London: Longmans, Green & Co, 1932.

[79] C. Hammann, A. Luptak, J. Perreault, and M. de la Peña. The ubiquitous hammerhead ribozyme. *RNA*, 18(5):871–885, 2012.

[80] E.J. Hayden, E. Ferrada, and A. Wagner. Cryptic genetic variation promotes rapid evolutionary adaptation in an RNA enzyme. *Nature*, 474(7349):92–95, 2011.

[81] I.L. Hofacker, W. Fontana, P.F. Stadler, L.S. Bonhoeffer, M. Tacker, and P. Schuster. Fast folding and comparison of rna secondary structures. *Monatshefte für Chemie/Chemical Monthly*, 125(2):167–188, 1994.

[82] I.L. Hofacker, P. Schuster, and P.F. Stadler. Combinatorics of RNA secondary structures. *Discrete Applied Mathematics*, 88(1):207–237, 1998.

[83] W. Huang, J. Petrosino, M. Hirsch, P.S. Shenkin, and T. Palzkill. Amino acid sequence determinants of [beta]-lactamase structure and activity. *Journal of molecular biology*, 258(4):688–703, 1996.

[84] L.D. Hurst. Genetics and the understanding of selection. *Nature Reviews Genetics*, 10(2):83–93, 2009.

[85] S. Itzkovitz and U. Alon. The genetic code is nearly optimal for allowing additional information within protein-coding sequences. *Genome research*, 17(4):405, 2007.

[86] W. Johannsen. The genotype conception of heredity. *The American Naturalist*, 45(531):129–159, 1911.

[87] W. Johannsen. *Elemente der exakten Erblichkeitslehre*. Fischer Jena, 1909.

[88] I.G. Johnston. *Exploration, Exploitation and Complexity in Biological Evolution and Self Assembly*. PhD thesis, University of Oxford, 2010.

[89] I.G. Johnston, K. Dingle, S.F. Greenbury, J.P.K. Doye, S.E. Ahnert, and A.A. Louis. Symmetry and modularity spontaneously arise from the algorithmic nature of evolution. *In preparation*, 2014.

[90] I.G. Johnston, S.A. Ahnert, J.P.K. Doye, and A.A. Louis. Evolutionary dynamics in a simple model of self-assembly. *Physical Review E*, 83:066105, 2011.

[91] T. Jorg, O.C. Martin, and A. Wagner. Neutral network sizes of biological RNA molecules can be computed and are not atypically small. *BMC bioinformatics*, 9(1):464, 2008.

[92] F. Kaspar and H.G. Schuster. Easily calculable measure for the complexity of spatiotemporal patterns. *Physical Review A*, 36(2):842, 1987.

[93] S.A. Kauffman. Metabolic stability and epigenesis in randomly constructed genetic nets. *Journal of theoretical biology*, 22(3):437–467, 1969.

[94] S.A. Kauffman. *The origins of order*. Oxford University Press New York, 1993.

[95] S.A. Kauffman. *At home in the universe: The search for laws of self-organization and complexity*. Oxford University Press, USA, 1995.

[96] A.D. Keefe and J.W. Szostak. Functional proteins from a random-sequence library. *Nature*, 410(6829):715–717, 2001.

[97] B.S. Khatri, T.C.B. McLeish, and R.P. Sear. Statistical mechanics of convergent evolution in spatial patterning. *Proceedings of the National Academy of Sciences*, 106(24):9564, 2009.

[98] M. Kimura. *The neutral theory of molecular evolution*. Cambridge University Press, 1985.

[99] T. Kin, K. Yamada, G. Terai, H. Okida, Y. Yoshinari, Y. Ono, A. Kojima, Y. Kimura, T. Komori, and K. Asai. fRNAdb: a platform for mining/annotating functional RNA candidates from non-coding RNA sequences. *Nucleic acids research*, 35(suppl 1):D145–D148, 2007.

[100] J.L. King and T.H. Jukes. Non-darwinian evolution. *Science (New York, NY)*, 164(881):788, 1969.

[101] L.G. Kleina, J.H. Miller, et al. Genetic studies of the lac repressor. xiii. extensive amino acid replacements generated by the use of natural and synthetic nonsense suppressors. *Journal of molecular biology*, 212(2):295, 1990.

[102] R.D. Knight, S.J. Freeland, and L.F. Landweber. Rewiring the keyboard: evolvability of the genetic code. *Nature Reviews Genetics*, 2(1):49–58, 2001.

[103] R.D. Knight, S.J. Freeland, and L.F. Landweber. A simple model based on mutation and selection explains trends in codon and amino-acid usage and gc

composition within and across genomes. *Genome Biology*, 2(4):research0010, 2001.

[104] R. Knight, H. De Sterck, R. Markel, S. Smit, A. Oshmyansky, and M. Yarus. Abundance of correctly folded RNA motifs in sequence space, calculated on computational grids. *Nucleic acids research*, 33(18):5924–5935, 2005.

[105] R. Knight and M. Yarus. Finding specific RNA motifs: Function in a zeptomole world? *RNA*, 9(2):218–230, 2003.

[106] A.N. Kolmogorov. Three approaches to the quantitative definition of information. *Problems of information transmission*, 1(1):1–7, 1965.

[107] E.V. Koonin and A.S. Novozhilov. Origin and evolution of the genetic code: the universal enigma. *IUBMB life*, 61(2):99–111, 2009.

[108] K.R. Lang and O. Gingerich. A source book in astronomy and astrophysics, 1900-1975. 1979.

[109] A. Lempel and J. Ziv. On the complexity of finite sequences. *Information Theory, IEEE Transactions on*, 22(1):75–81, 1976.

[110] L.A. Levin. Laws of information conservation (nongrowth) and aspects of the foundation of probability theory. *Problemy Peredachi Informatsii*, 10(3):30–35, 1974.

[111] E.D. Levy, J.B. Pereira-Leal, C. Chothia, and S.A. Teichmann. 3D complex: a structural classification of protein complexes. *PLoS computational biology*, 2(11):e155, 2006.

[112] R. Lewontin. The genotype/phenotype distinction. In Edward N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Summer 2011 edition, 2011.

[113] F. Li, T. Long, Y. Lu, Q. Ouyang, and C. Tang. The yeast cell-cycle network is robustly designed. *Proceedings of the National Academy of Sciences* , 101(14):4781, 2004.

[114] H. Li, R. Helling, C. Tang, and N. Wingreen. Emergence of preferred structures in a simple model of protein folding. *Science*, 273(5275):666, 1996.

[115] H. Li, C. Tang, and N.S. Wingreen. Designability of protein structures: A lattice-model study using the miyazawa-jernigan matrix. *Proteins: Structure, Function, and Bioinformatics*, 49(3):403–412, 2002.

[116] M. Li, J.H. Badger, X. Chen, S. Kwong, P. Kearney, and H. Zhang. An information-based sequence distance and its application to whole mitochondrial genome phylogeny. *Bioinformatics*, 17(2):149–154, 2001.

[117] M. Li and P.M.B. Vitanyi. *An introduction to Kolmogorov complexity and its applications*. Springer-Verlag New York Inc, 2008.

[118] W.-H. Li. *Molecular evolution.* Sinauer Associates Incorporated, 1997.

[119] A. Lindenmayer. Mathematical models for cellular interactions in development i. filaments with one-sided inputs. *Journal of theoretical biology*, 18(3):280–299, 1968.

[120] D.J. Lipman and W.J. Wilbur. Modelling neutral and selective evolution of protein folding. *Proceedings of the Royal Society of London. Series B: Biological Sciences*, 245(1312):7–11, 1991.

[121] L. Loewe. A framework for evolutionary systems biology. *BMC Systems Biology*, 3(1):27, 2009.

[122] M. Lynch. The frailty of adaptive hypotheses for the origins of organismal complexity. *Proceedings of the National Academy of Sciences*, 104(Suppl 1):8597, 2007.

[123] A.L. Mackay. Optimization of the genetic code. *Nature*, 216:159–160, 1967.

[124] J. Matoušek and J Nešetřil. *Invitation to Discrete Mathematics*. Oxford University Press, 2009.

[125] B.W. Matthews. Mutational analysis of protein stability: Current opinion in structural biology 1991, 1: 17–21. *Current Opinion in Structural Biology*, 1(1):17–21, 1991.

[126] T. McKee and J.R. McKee. *Biochemistry: the molecular basis of life*. Oxford University Press, 2009.

[127] T. McKee and J.R. McKee. *Biochemistry: the molecular basis of life*. Oxford University Press, 2010.

[128] M. Mezard and A. Montanari. *Information, physics, and computation*. Oxford University Press, USA, 2009.

[129] N.K. Mimouni, R.B. Lyngsø, S. Griffiths-Jones, and J. Hein. An analysis of structural influences on selection in RNA genes. *Molecular biology and evolution*, 26(1):209–216, 2009.

[130] M. Mitchell. *Complexity: a guided tour*. Oxford University Press, 2009.

[131] J. Monod, J. Wyman, and J.-P. Changeux. On the nature of allosteric transitions: a plausible model. *Journal of Molecular Biology*, 12:228–118, 1965.

[132] A. Muto and S. Osawa. The guanine and cytosine content of genomic DNA and bacterial evolution. *Proceedings of the National Academy of Sciences*, 84(1):166, 1987.

[133] N. Nagano, C.A. Orengo, and J.M. Thornton. One fold with many functions: The evolutionary relationships between tim barrel families based on their sequences, structures and functions. *Journal of Molecular Biology*, 321(5):741–765, 2002.

[134] I. Nobeli, A.D. Favia, and J.M. Thornton. Protein promiscuity and its implications for biotechnology. *Nature biotechnology*, 27(2):157–167, 2009.

[135] D. Noble. Biophysics and systems biology. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 368(1914):1125, 2010.

[136] Y.D. Nochomovitz and H. Li. Highly designable phenotypes and mutational buffers emerge from a systematic mapping between network topology and dynamic output. *Proceedings of the National Academy of Sciences*, 103(11):4180, 2006.

[137] M. Pavlicev, J.M. Cheverud, and G.P. Wagner. Evolution of adaptive phenotypic variation patterns by direct selection for evolvability. *Proceedings of the Royal Society B: Biological Sciences*, 278(1713):1903–1912, 2011.

[138] I.S. Peter, E. Faure, and E.H. Davidson. Predictive computation of genomic logic processing functions in embryonic development. *Proceedings of the National Academy of Sciences*, 109(41):16434, 2012.

[139] M. Peto, A. Kloczkowski, and R.L. Jernigan. Shape-dependent designability studies of lattice proteins. *Journal of Physics: Condensed Matter*, 19:285220, 2007.

[140] T. Piersma and J.A. Van Gils. *The flexible phenotype: a body-centred integration of ecology, physiology, and behaviour*. Oxford University Press Oxford, 2011.

[141] M. Pigliucci. Genotype–phenotype mapping and the end of the genes as blueprint metaphor. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 365(1540):557, 2010.

[142] M. Pigliucci and G. Müller. *Evolution - the extended synthesis*. MIT Press, 2010.

[143] M. Pigliucci. *Phenotypic plasticity: beyond nature and nurture*. Johns Hopkins University Press, 2001.

[144] R. Piskol and W. Stephan. Selective constraints in conserved folded RNAs of drosophilid and hominid genomes. *Molecular biology and evolution*, 28(4):1519–1529, 2011.

[145] S. Psujek and R.D. Beer. Developmental bias in evolution: evolutionary accessibility of phenotypes in a model evo-devo system. *Evolution & Development*, 10(3):375–390, 2008.

[146] K. Raman and A. Wagner. Evolvability and robustness in a complex signalling circuit. *Molecular BioSystems*, 7:1081–1092, 2011.

[147] K. Raman and A. Wagner. The evolvability of programmable hardware. *Journal of The Royal Society Interface*, 8(55):269, 2011.

[148] D. Rennell, S.E. Bouvier, L.W. Hardy, A.R. Poteete, et al. Systematic mutation of bacteriophage t4 lysozyme. *Journal of molecular biology*, 222(1):67, 1991.

[149] R.C. Richmond. Non-darwinian evolution: a critique. *Nature*, 225:1025–1028, 1970.

[150] M. Ridley. *Evolution, 3rd edition*. Blackwell Science, 2004.

[151] K.F. Riley, M.P. Hobson, and S.J. Bence. *Mathematical methods for physics and engineering*. Cambridge University Press, 2006.

[152] E. Rivals, O. Delgrange, J.P. Delahaye, M. Dauchet, M.O. Delorme, A. Hénaut, and E. Ollivier. Detection of significant patterns by compression algorithms: the case of approximate tandem repeats in DNA sequences. *Computer applications in the biosciences: CABIOS*, 13(2):131–136, 1997.

[153] K. Salehi-Ashtiani and J.W. Szostak. In vitro evolution suggests multiple origins for the hammerhead ribozyme. *Nature*, 414(6859):82–83, 2001.

[154] R. Sanjuán, J.M. Cuevas, V. Furió, E.C. Holmes, and A. Moya. Selection for robustness in mutagenized RNA viruses. *PLoS genetics*, 3(6):e93, 2007.

[155] S. Schaper, I.G. Johnston, and A.A. Louis. Epistasis can lead to fragmented neutral spaces and contingency in evolution. *Proceedings of the Royal Society B: Biological Sciences*, 279(1734):1777–1783, 2012.

[156] S. Schaper and A.A. Louis. The arrival of the frequent: How bias in genotype-phenotype maps can steer populations to local optima. *PloS one*, 9(2):e86635, 2014.

[157] E.A. Schultes, P.T. Hraber, and T.H. LaBean. Estimating the contributions of selection and self-organization in RNA secondary structure. *Journal of Molecular Evolution*, 49(1):76–83, 1999.

[158] E.A. Schultes, A. Spasic, U. Mohanty, and D.P. Bartel. Compact and ordered collapse of randomly generated RNA sequences. *Nature structural & molecular biology*, 12(12):1130–1136, 2005.

[159] P. Schuster. Landscapes and molecular evolution. *Physica D: Nonlinear Phenomena*, 107(2-4):351–365, 1997.

[160] P. Schuster, W. Fontana, P.F. Stadler, and I.L. Hofacker. From sequences to shapes and back: A case study in RNA secondary structures. *Proceedings: Biological Sciences*, 255(1344):279–284, 1994.

[161] G. Sella and A.E. Hirsh. The application of statistical physics to evolutionary biology. *Proceedings of the National Academy of Sciences*, 102(27):9541, 2005.

[162] J.P. Sethna. *Statistical mechanics: entropy, order parameters, and complexity*, volume 14. Oxford University Press, USA, 2006.

[163] B.E. Shakhnovich, E. Deeds, C. Delisi, and E. Shakhnovich. Protein structure and evolutionary history determine sequence space topology. *Genome research*, 15(3):385, 2005.

[164] M. Shionyu, K. Takahashi, and M. Gō. Variable subunit contact and cooperativity of hemoglobins. *Journal of Molecular Evolution*, 53(4):416–429, 2001.

[165] S. Smit, M. Yarus, and R. Knight. Natural selection is not required to explain universal compositional patterns in rRNA secondary structure categories. *RNA*, 12(1):1–14, 2006.

[166] J.M. Smith, R. Burian, S. Kauffman, P. Alberch, J. Campbell, B. Goodwin, R. Lande, D. Raup, and L. Wolpert. Developmental constraints and evolution: a perspective from the mountain lake conference on development and evolution. *Quarterly Review of Biology*, pages 265–287, 1985.

[167] R. J. Solomonoff. A preliminary report on a general theory of inductive inference (revision of report v-131). *Contract AF*, 49(639):376, 1960.

[168] P.R. Stein and M.S. Waterman. On some new sequences generalizing the Catalan and Motzkin numbers. *Discrete Mathematics*, 26(3):261–272, 1979.

[169] T.A. Steitz. DNA polymerases: structural diversity and common mechanisms. *Journal of Biological Chemistry*, 274(25):17395, 1999.

[170] D.L. Stern and V. Orgogozo. Is genetic evolution predictable? *Science*, 323(5915):746, 2009.

[171] A. Stoltzfus and L.Y. Yampolsky. Climbing mount probable: mutation as a cause of nonrandomness in evolution. *Journal of Heredity*, 100(5):637, 2009.

[172] Z. Sükösd, B. Knudsen, J.W.J. Anderson, Á. Novák, J. Kjems, and C.N.S. Pedersen. Characterising RNA secondary structure space using information entropy. *BMC bioinformatics*, 14(Suppl 2):S22, 2013.

[173] D.M. Taverna and R.A. Goldstein. The distribution of structures in evolving protein populations. *Biopolymers*, 53(1):1–8, 2000.

[174] S.V. Taylor, K.U. Walter, P. Kast, and D. Hilvert. Searching sequence space for protein catalysts. *Proceedings of the National Academy of Sciences*, 98(19):10596, 2001.

[175] P. Tina, A.M. Joseph, L.S. Filipa, N. Eviatar, J.C. Lucy, E.A. Sebastian, and A.T. Sarah. The emergence of protein complexes: quaternary structure, dynamics and allostery. *Biochemical Society Transactions*, 40(3):475–491, 2012.

[176] A.E. Tsong, B.B. Tuch, H. Li, and A.D. Johnson. Evolution of alternative transcriptional circuits with identical logic. *Nature*, 443(7110):415–420, 2006.

[177] R.L. Tuinstra, F.C. Peterson, S. Kutlesa, E.S. Elgin, M.A. Kron, and B.F. Volkman. Interconversion between two unrelated protein folds in the lymphotactin native state. *Proceedings of the National Academy of Sciences*, 105(13):5057, 2008.

[178] A.M. Turing. On computable numbers, with an application to the entscheidungsproblem. a correction. *Proceedings of the London Mathematical Society*, 2(1):544, 1938.

[179] A.J. Venkatakrishnan, E.D. Levy, and S.A. Teichmann. Homomeric protein complexes: evolution and assembly. *Biochemical Society Transactions*, 38(4):879, 2010.

[180] J.M.G. Vilar, H.Y. Kueh, N. Barkai, and S. Leibler. Mechanisms of noise-resistance in genetic oscillators. *Proceedings of the National Academy of Sciences*, 99(9):5988, 2002.

[181] P.M.B. Vitányi. Similarity and denoising. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 371(1984), 2013.

[182] A. Wagner. *Robustness and evolvability in living systems*. Princeton University Press Princeton, NJ:, 2005.

[183] A. Wagner. Neutralism and selectionism: a network-based reconciliation. *Nature Reviews Genetics*, 9(12):965–974, 2008.

[184] A. Wagner. Robustness and evolvability: a paradox resolved. *Proceedings of the Royal Society B*, 275(1630):91, 2008.

[185] A. Wagner. *The Origins of Evolutionary Innovations: A Theory of Transformative Change in Living Systems*. Oxford University Press, 2011.

[186] A. Wagner, and P.F. Stadler Viral RNA and evolved mutational robustness. *Journal of Experimental Zoology*, 285(2):119–127, 1999.

[187] T. Warnecke, C.C. Weber, and L.D. Hurst. Why there is more to protein evolution than protein function: splicing, nucleosomes and dual-coding sequence. *Biochemical Society Transactions*, 37(4):756, 2009.

[188] C.O. Wilke. Molecular clock in neutral protein evolution. *BMC genetics*, 5(1):25, 2004.

[189] C.O. Wilke. Selection for fitness versus selection for robustness in rna secondary structure folding. *Evolution*, 55(12):2412–2420, 2001.

[190] K. Willbrand, F. Radvanyi, J.P. Nadal, J.P. Thiery, and T.M.A. Fink. Identifying genes from up–down properties of microarray expression series. *Bioinformatics*, 21(20):3859–3864, 2005.

[191] C.R. Woese and N. Goldenfeld. How the microbial world saved evolution from the scylla of molecular biology and the charybdis of the modern synthesis. *Microbiology and molecular biology reviews*, 73(1):14–21, 2009.

[192] R.J. Woods, J.E. Barrick, T.F. Cooper, U. Shrestha, M.R. Kauth, and R.E. Lenski. Second-order selection for evolvability in a large escherichia coli population. *Science*, 331(6023):1433, 2011.

[193] S. Wright. The roles of mutation, inbreeding, crossbreeding and selection in evolution. In *Proceedings of the 6th International Congress of Genetics*, volume 1, pages 356–366, 1932.

[194] R. Wroe, E. Bornberg-Bauer, and H.S. Chan. Comparing folding codes in simple heteropolymer models of protein evolutionary landscape: robustness of the superfunnel paradigm. *Biophysical journal*, 88(1):118–131, 2005.

[195] J.-Yi. Yang, Z. Yu, and V. Anh. Correlations between designability and various structural characteristics of protein lattice models. *Journal of Chemical Physics*, 126(19):1–12, 2007.

[196] K. Yue and K.A. Dill. Forces of tertiary structural organization in globular proteins. *Proceedings of the National Academy of Sciences*, 92(1):146, 1995.

[197] H. Zenil and J.P. Delahaye. An algorithmic information theoretic approach to the behaviour of financial markets. *Journal of Economic Surveys*, 25(3):431–463, 2011.

[198] H. Zenil, F. Soler-Toscano, J.-P. Delahaye, and N. Gauvrit. Two-dimensional Kolmogorov complexity and validation of the coding theorem method by compressibility. *arXiv preprint arXiv:1212.6745*, 2012.

[199] H. Zenil, F. Soler-Toscano, K. Dingle, and A.A. Louis. Correlation of automorphism group size and topological properties with program-size complexity evaluations of graphs and complex networks. *Physica A: Statistical Mechanics and its Applications*, 404:341–358, 2014.

[200] T. Zhang, R. Ramakrishnan, and M. Livny. BIRCH: an efficient data clustering method for very large databases. In *ACM SIGMOD Record*, volume 25, pages 103–114. ACM, 1996.

[201] M. Zuker, D.H. Mathews, and D.H. Turner. Algorithms and thermodynamics for RNA secondary structure prediction: a practical guide. *RNA biochemistry and biotechnology*, 70:11–44, 1999.