A Striking Property of Genetic Code-Like Transformations

Hillol Kargupta
Department of Computer Science and Electrical Engineering
University of Maryland Baltimore County
Baltimore, MD 21250, USA
hillol@cs.umbc.edu

The gene expression process in nature plays a key role in evaluating the fitness of DNA through the production of different proteins in different cells. The production of proteins from DNA goes through different stages. Among others, the transcription stage produces the mRNA from the DNA and translation produces the amino acid sequence in protein from the mRNA. The translation process is accomplished by mapping the mRNA sequence using a transformation called the *genetic code*. This code considers every consequent triplet (codon) of nucleic acids in the mRNA sequence and maps it to a corresponding amino acid. This paper shows that genetic code-like transformations introduce very interesting properties to the representation of a genetic fitness function. It presents a Fourier[1] analysis of genetic code-like transformations. It points out that such transformations can convert some function representations of exponential description in Fourier basis to a description that is highly suitable for polynomial complexity approximation. More precisely, such transformations can construct a Fourier representation with only a polynomial number of terms that are exponentially more significant than the rest. Polynomial-complexity approximation of functions from data is a fundamental problem in inductive learning, data mining, search, and optimization. Therefore the work has important implications in these areas. It is unlikely that such representations can be constructed for all functions. However, since such transformations appear to work well in nature, the class of such functions may not be trivial and we should explore it further.

Key Words: Gene expression, genetic code, function induction.

[1]The analysis is identical to that using Walsh basis [4, 45]; however, the the term Fourier is chosen because of its historical [32, 18] use in function approximation literature.

1. Introduction

Learning functions from data is important in many fields such as inductive learning, statistics, and data mining. It may also be important for non-enumerative optimization in absence of sufficient domain knowledge; this is because such optimization may require inductive detection of structure of the objective function for intelligent guessing about the desired solution.

Representation plays an important role in learning functions. For example, if the function has an exponentially large (in the number of variables defining the function) description in the chosen representation then its polynomial-time computation is not possible. On the other hand, a representation with polynomially bounded size is amenable to efficient computation. A function with an exponentially large description may be efficiently computed when it can be approximated using a function that has a polynomially bounded description size. This may be possible when the target function has an exponentially large representation with only a polynomial number of "significant" components. In that case, we may be able to neglect the "insignificant" components and still enjoy a high degree of accuracy. Therefore, constructing function representations with a "small" number of "significant" components is important for efficient function induction.

This paper considers Fourier basis representation of some well known functions and shows that there exists a class of transformations that offers this property in the Fourier space under some practical conditions. The transformations are similar to the genetic code that transforms the genetic fitness function defined over the protein sequences to the mRNA representation in a living organism.

A living body starts its life from the DNA, the primary information career in genetics. Almost every critical activity of the organism is accomplished by proteins constructed from the DNA. The efficacy of the organism, i.e. the genetic fitness, depends on the proteins. For some reason our body chooses different representations of the information stored in the proteins. It uses the mRNA and the DNA sequences to represent the proteins. It first transforms the DNA to the mRNA representation and subsequently to the protein before evaluating the fitness of the genome. This process of representation transformations is called gene expression. Representation transformations are often used in many fields like Physics, Engineering, Machine Learning, and Mathematics for transforming difficult problems into suitable forms that are easier to solve. Therefore representation transformations in gene expression allude intriguing possibilities.

This paper investigates the possible role of the gene expression in making genetic search efficient. It considers one important part of gene expression, the translation, that transforms the mRNA sequence to protein. Translation is governed by the *genetic code*. This paper presents a Fourier analysis of genetic code-like transformations in the binary sequence space and demonstrates a quite interesting property of such representation transformations. It points out that there exists some genetic code-like transformations that can convert some functions with exponentially long description in Fourier basis to a representation where only a polynomial number of terms are exponentially more significant than the rest when fitter proteins are given

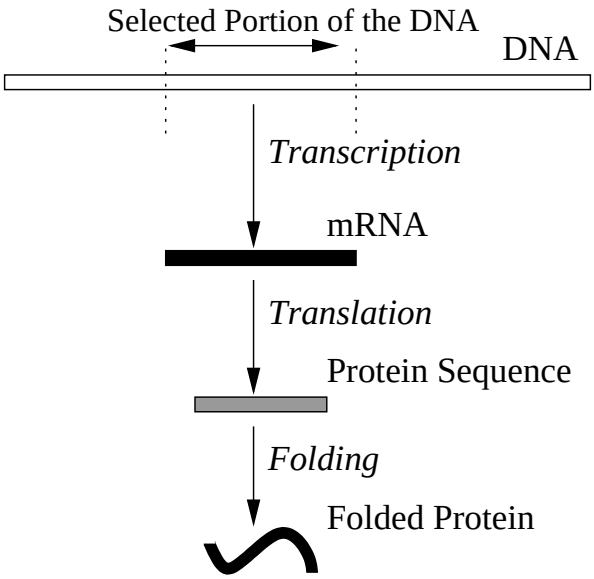

Figure 1. Different steps of gene expression.

more copies through redundant and equivalent representation.

Section 2 describes the gene expression process in nature. Section 3 briefly reviews the previous work on the computation in gene expression. Section 4 reviews the basics of Fourier representation. Section 5 analyzes the effect of genetic code-like transformations on the representation of the genetic fitness function and proves the main results of this paper. Finally Section 6 concludes this paper.

2. Gene Expression and the Genetic Code

The DNA is the primary carrier of the genetic information that is transmitted from one generation to another. DNA molecules consist of two long complementary chains held together by base pairs. DNA consists of four kinds of bases joined to a sugar-phosphate backbone. The four bases in DNA are *adenine* (A), *guanine* (G), *thymine* (T) and *cytosine* (C). Chromosomes are made of DNA *double helices*. Bases in DNA helices obey the *complementary base pairing rule*. T and G pair with A and C respectively. In other words, if the base at a particular position of a helix is T then the corresponding base in the other helix should be A. The information coded in the DNA is extracted during the process of gene expression.

Expression of genetic information coded in DNA requires construction of the mRNA sequence, followed by that of proteins. The main steps are,

- transcription: formation of mRNA (messenger ribonucleic acid) from DNA
- translation: formation of protein from mRNA

Protein feature	mRNA codons
Alanine	GCA GCC GCG GCU
Cysteine	UGC UGU
Aspartic acid	GAC GAU
Glutamic acid	GAA GAG
Phenylalanine	UUC UUU
Glycine	GGA GGC GGG GGU
Histidine	CAC CAU
Isoleucine	AUA AUC AUU
Lysine	AAA AAG
Leucine	UUA UUG CUA CUC CUG CUU
Methionine	AUG
Asparagine	AAC AAU
Proline	CCA CCC CCG CCU
Glutamine	CAA CAG
Arginine	AGA AGG CGA CGC CGG CGU
Serine	AGC AGU UCA UCC UCG UCU
Threonine	ACA ACC ACG ACU
Valine	GUA GUC GUG GUU
Tryptophan	UGG
Tyrosine	UAC UAU
STOP	UAA UAG UGA

Table 1. The universal genetic code.

- protein folding

In a particular cell, transcription produces the mRNA from a small portion of the DNA. The mRNA defines another level of representation of the genetic information. It consists of four types of bases joined to a ribose-sugar-phosphodiester backbone. The four bases are *adenine* (A), *uracil* (U), *guanine* (G), and *cytosine* (C). All the bases defining the mRNA are same as those in DNA sequences, except that T is replaced by U. The mRNA is produced from the DNA by RNA Polymerase and the regulatory proteins following the *complementary base-pairing rules* similar to those in DNA. The RNA Polymerase initiates the transcription at a place of the DNA marked by the *promoter* region (*start site*). It splits the DNA double helix and continues generating the mRNA using one of the DNA strands as a template. The RNA Polymerase stops when it finds a termination signal sequence (*stop site*)in the DNA strand. Note that only a small portion of the DNA strand is transcribed and different cells may transcribe different regions of the DNA for producing proteins.

The mRNA acts as the template for protein synthesis. A protein is defined by a sequence of *amino acids*, joined by peptide bonds. The mRNA is transported to the cell cytoplasm for producing protein in the ribosome. There exists a set of rules that defines the correspondence between nucleotide triplets (known as *codons*) and the amino acids in proteins. This is known as the *genetic code*. Each codon is comprised of three adjacent nucleotides in a DNA chain and it produces a unique amino

acid. With a few exceptions the genetic code for most eukaryotic and prokaryotic organisms is the same. Amino acid sequence defines a new representation of the information coded in mRNA.

The final level of representation of genetic information is defined by the three dimensional structure of folded proteins. Although amino acid sequences fundamentally define proteins, formation of the three dimensional structure of proteins involves a complex process, often called *protein folding*. This process involves interaction between multiple amino acid subsequences, resulting in the emergence in a folded structure from the sequence.

Figure 1 shows the different steps of the gene expression process. Although proteins play a key role in determining the genetic fitness, the purpose of the two additional layers of representations (mRNA and DNA) for representing the fitness function is not clear. Representation transformations are often used in physics, engineering, and machine learning for solving problems efficiently. Therefore, the role of gene expression in efficient genetic search is really intriguing. This paper investigates gene expression from this perspective. First, let us review the existing related literature.

3. Previous Work

The importance of gene expression in genetic search was realized in the early days of the field of genetic algorithms. Holland [15] described the dominance operator as a possible way to model the effect of gene expression in diploid chromosomes. He also noted the importance of the process of protein synthesis from DNA in the computational model of genetics. Despite the fact that, traditionally dominance maps are explained from the Mendelian perspective, Holland made an interesting leap by connecting it to the synthesis of protein by gene signals, which today is universally recognized as gene expression. He realized the relation between the dominance operator with the "operon" model of the functioning of the chromosome [19] in evolution and pointed out the possible computational role of gene signaling in evolution [15].

Several other efforts have been made to model some aspects of gene expression. Diploidy and dominance have also been used elsewhere [1, 7, 16, 38, 40]. Most of them took their inspiration from the Mendelian view of genetics. The under-specification and over-specification decoding operator of messy GA has been viewed as a mechanism similar to gene signaling [13]. The structured genetic algorithm [8] also shares motivations from the gene expression; it uses a structured hierarchical representation in which genes are collectively switched on and off. This provides the search algorithm with a richer representation and helps capturing properties of the landscape better. An empirical study of genetic programming using artificial genetic code is presented in [31]. Kauffman [30] offered an interesting perspective of the natural evolution that realizes the importance for gene expression. However, Kauffman's work does not explain the process in basic computational terms on analytical grounds and does not relate the issue to the complexity of search. The complex nature of the representation in the DNA itself created interest among the

researchers. The eukaryotic DNA typically contains many segments that are not used in the gene expression process for producing proteins. An empirical investigation of the role of such "non-coding" segments (introns) in genetic search can be found in [46]. A survey of evolutionary algorithms with intron-based representations is presented in [47].

The "neutral network" theory [36, 39] also considers sequence-to-structure mapping from the perspective of random graph construction. This work approaches gene expression from the perspective of random graph construction and points out existence of the fitness invariant neutral networks. The translation process maps multiple mRNA sequences to the same protein sequence. As a result, it creates a genetic space that contains multiple genomes with same fitness, termed as neutral networks. This work provides interesting insights into the effect of such neutral networks in genetic search. However, its contribution towards polynomial-time representation construction of genetic fitness function is not clear.

Another related effort to understand the properties of the fitness landscape defined by the mRNA can be found in [37]. This work presents a Fourier analysis of the landscapes derived from the RNAs using Fast Fourier Transformation (FFT). Although the time complexity of the FFT is better than the regular Fourier Transformation, it still grows exponentially with respect to the number of feature variables defining the domain of the fitness function. This paper suggests that the genetic code that transforms the RNA to protein itself may help designing a polynomial time algorithm for the construction of the Fourier representation which the FFT cannot offer.

There also exists a body of literature that investigates the evolution of the genetic code. An algebraic model of the evolution of the genetic code is presented in [17]. This work searches for symmetries in the genetic code and points out the existence of a unique approximate symmetry group compatible with the codon assignments. The main idea behind this work is to view the evolution of the genetic code as an iterative process of representation decomposition. The genetic code is viewed as a 64-dimensional representation decomposed into several sub-representations with respect to different subgroups. The number of amino acids correspond to the number of sub-representations and the number of codons for any amino acid corresponds to the dimension of that subrepresentation. An extension of this work using Lie superalgebra is presented in [3]. Additional work on the different biological theories on the evolution of the genetic code can be found elsewhere [5, 10].

An alternate approach has been developed by Kargupta and his colleagues [2, 20, 29, 21, 22, 26, 23, 25, 24, 28]. This approach is mainly motivated by a perspective of the gene expression as a mechanism to make genetic search more efficient. This approach notes that the traditional model of evolutionary computation (based on selection, crossover, and mutation)[15] appears to have some serious scalability problems [42] for reasonably difficult problems. There is also little theoretical result available that proves guaranteed polynomial time performance of existing evolutionary algorithms for reasonably difficult classes of problems. Since the existing models of evolutionary computation do not address the gene expression issue very well and gene expression changes the genetic representation, it may become

a natural candidate for exploring the unknown mechanism that makes the genetic search in nature so efficient and scalable.

The early exploration of gene expression-like mechanisms for efficient inductive detection of function structure resulted in a class of heuristics-based techniques, known as the gene expression messy GA (GEMGA) [22]. In the recent past, more rigorous approaches using Fourier basis representations are suggested. Fourier Representations exposes the underlying function structure and it is functionally complete. Therefore, if we can learn such representations quickly, the purpose of function induction is served. A randomized algorithm is presented in [28, 27] that can induce a representation in Fourier basis in polynomial time for problems with bounded variable interaction (BVI). The assumption of BVI makes sure that among ℓ features defining the search domain, only at most some k (a constant) number of variables can interact with each other. In other words, the overall fitness function can be decomposed into a collection of either overlapping or non-overlapping sub-functions where each of the sub-functions can depend on at most k variables. This condition guarantees a polynomial size description of the target function in Fourier representation. An alternate technique for estimating the Fourier representations is proposed elsewhere [18, 32]. An extension of this technique for detecting function structure in genetic algorithms is reported in [41].

Although there exists many functions with a polynomial-size canonical representation, it is not clear why the natural genetic fitness function should have such a property. This paper suggests a possible direction to answer this question. It shows that genetic code-like transformations can construct a Fourier representations of at least some fitness functions where the contribution of Fourier coefficients involving some q features decreases exponentially with q. This may allow us to approximate the genetic fitness function with a Fourier representation that neglects the effect of Fourier coefficients associated with some k or higher features. In other words the approximation will satisfy the BVI property. If that is the case, then we can induce such functions efficiently in polynomial time. The following section reviews the fundamentals of Fourier representation.

4. Fourier Representation and Function Induction

The role of the genetic code in the evaluation of the fitness can be understood in the context of an appropriately chosen set of basis functions. This paper uses the Fourier basis functions to do that. The representation is very similar to the Walsh basis [4, 45], frequently used by the genetic algorithm community. How this paper uses the Fourier representation because of its history in function induction literature [32, 18]. The following section presents a brief review of the Fourier basis and its relation with the problem of inducing functions from data.

4.1 A brief review of the Fourier basis

Fourier bases are orthogonal functions that can be used to represent any function. In this paper we shall consider functions of binary variables. Consider the function

space over the set of all ℓ-bit strings. The Fourier basis set that spans this space is comprised of 2^ℓ functions. Each Fourier basis function is defined as follows:

$$\psi_{\mathbf{j}}(\mathbf{x}) = (-1)^{(\mathbf{x}\cdot\mathbf{j})} \tag{1}$$

Where $\mathbf{j}$ and $\mathbf{x}$ are binary strings of length ℓ. In other words $\mathbf{j} = j_1, j_2, \cdots j_\ell$, $\mathbf{x} = x_1, x_2, \cdots x_\ell$ and $\mathbf{j}, \mathbf{x} \in \{0,1\}^\ell$; $\mathbf{x}\cdot\mathbf{j}$ denotes the inner product of $\mathbf{x}$ and $\mathbf{j}$ which is nothing but $\sum_{i=1}^{\ell} x_i j_i$. $\psi_{\mathbf{j}}(\mathbf{x})$ can either be equal to 1 or -1. The string $\mathbf{j}$ is called a *partition*. The *order* of a partition $\mathbf{j}$ is the number of 1-s in $\mathbf{j}$. A Fourier basis function depends on some x_i only when $j_i = 1$. Therefore a partition can also be viewed as a representation of a certain subset of x_i-s; every unique partition corresponds to a unique subset of x_i-s. If a partition $\mathbf{j}$ has exactly α number of 1-s then we say the partition is of order α since the corresponding Fourier function depends on only those α number of variables corresponding to the 1-s in the partition $\mathbf{j}$. Fourier bases are orthonormal. Therefore,

$$\begin{aligned} \frac{1}{2^\ell}\sum_{\mathbf{x}} \psi_{\mathbf{i}}(\mathbf{x})\psi_{\mathbf{j}}(\mathbf{x}) &= 1 \quad \text{when} \quad \mathbf{i} = \mathbf{j} \\ &= 0 \quad \text{when} \quad \mathbf{i} \neq \mathbf{j} \end{aligned}$$

A function $f : \mathbf{X}^\ell \to \Re$, that maps an ℓ-dimensional space of binary strings to a real-valued range, can be represented using the Fourier basis functions.

$$f(\mathbf{x}) = \sum_{\mathbf{j}} w_{\mathbf{j}} \psi_{\mathbf{j}}(\mathbf{x}) \tag{2}$$

where $w_{\mathbf{j}}$ is the Fourier Coefficient (FC) corresponding to the partition $\mathbf{j}$.

$$w_{\mathbf{j}} = \frac{1}{2^\ell}\sum_{\mathbf{x}} f(\mathbf{x})\psi_{\mathbf{j}}(\mathbf{x}) \tag{3}$$

We note from Equation 2 that a function can be expressed as a linear sum of the Fourier functions, each weighed by the corresponding Fourier coefficient. The Fourier coefficient $w_{\mathbf{j}}$ can be viewed as the relative contribution of the partition $\mathbf{j}$ to the function value of $f(\mathbf{x})$. Therefore, the absolute value of $w_{\mathbf{j}}$ can be used as the "significance" of the corresponding partition $\mathbf{j}$. If the magnitude of some $w_{\mathbf{j}}$ is very small compared to other coefficients then we may consider the $\mathbf{j}$-th partition to be insignificant and neglect its contribution.

Fourier bases and their close relatives Walsh bases are frequently used to study the behavior of genetic algorithms. Walsh bases [4] were first used by Bethke [6] for analyzing genetic algorithms. Further investigation of this approach can be found elsewhere [9, 11, 12, 14, 33, 34, 35, 43, 44].

4.2 Function induction from data and Fourier basis

Function induction from data plays an important role in adaptation, machine learning, and non-enumerative black-box optimization. In function induction, the goal is to learn a function $\hat{f} : X^\ell \to Y$ from the data set $\Omega = \{(\mathbf{x}_{(1)}, \mathbf{y}_{(1)}), (\mathbf{x}_{(2)}, \mathbf{y}_{(2)}), \cdots$

$(\mathbf{x}_{(k)}, \mathbf{y}_{(k)})\}$ generated by some underlying target function $f : X^n \rightarrow Y$, such that the $\hat{f}$ approximates f. Since Fourier basis is functionally complete, any function can represented in Fourier basis. Therefore, learning a function $\hat{f}$ can be posed as the problem of approximating the Fourier representation of f. If we can accurately estimate the significant coefficients (coefficients with relatively large magnitude) of the Fourier representation of f then we can use those coefficients to define $\hat{f}$. The complexity of inducing a function in Fourier representation is directly proportional to the number of such coefficients.

Note that the Fourier representation in the binary domain can potentially have 2^ℓ coefficients and estimating all of them will require an exponential time. However, we may be able to get away with polynomial time computation if there is a way to accurately approximate the function with an exponentially long description by a function with only a polynomially long description. For example, if the function has only a polynomial number of significant FCs then we may be able to construct an approximation by considering only those significant coefficients and neglecting the rest. In that case we can write $\hat{f} = \sum_{\mathbf{j}} w''_{\mathbf{j}} \psi_{\mathbf{j}}(\mathbf{x})$, where $w''_{\mathbf{j}} = w_{\mathbf{j}}$ when $|w_{\mathbf{j}}| \geq \theta$ and $w''_{\mathbf{j}} = 0$ otherwise. $|w_{\mathbf{j}}|$ denotes the magnitude of $w_{\mathbf{j}}$ and θ represents the chosen threshold. If the number of FCs in $\hat{f}$ is polynomially bound then its Fourier representation can be computed in polynomial time [18, 28, 27, 32, 41].

Unfortunately, function representations may not always come with such a nice property. The following section points out that there exists some genetic code-like transformations that can construct representations of functions with this very desirable property.

5. Exploring Genetic Code-like Transformations

The genetic code transforms the mRNA sequence to the protein sequence by assigning one protein feature for every codon in the mRNA sequence. Although the cardinalities of the alphabet sets of the mRNAs and proteins are more than two, understanding the underlying computation may require abstraction. In this section we will do so by assuming that the protein and the mRNA sequences are binary strings. Our objective is to explore the effect of the genetic code-like representation transformations in the binary domain using Fourier analysis. In order to do that first we need to define what we mean by genetic code-like transformations.

5.1 The notion of genetic code-like transformations

The genetic code defines the correspondence between an mRNA codon and a protein feature value. Although in nature the codons are defined by three mRNA feature values, the implication of the choice of number "three" is yet to be explained. Therefore, the current analysis will treat this as a parameter and the results of this paper can be specialized for any size of codons including three. As noted earlier, the analysis considers the effect of such transformations in the binary space. Although strings are binary, we will continue to use the terms mRNA, protein, and genetic code accordingly for maintaining the link between biology and the current analysis.

Protein feature	mRNA codon
1	100
1	000
1	001
1	010
0	111
0	101
0	110
0	011

Table 2. Code A: A genetic code-like transformation for binary representation. Single bit in the protein space maps to 3-bit codons in the mRNA space.

Protein feature	mRNA codon
0	100
0	000
0	001
0	010
0	111
0	101
0	110
1	011

Table 3. Code B: Another genetic code-like transformation for binary representation. Note that seven unique mRNA codons map to the protein feature value of zero.

Let us use $\mathbf{r}$ and $\mathbf{p}$ to represent the mRNA and the protein sequences respectively. Let ℓ_r and ℓ_p be their respective lengths. Just like the natural *translation* process, our artificial translation maps the mRNA sequence to the corresponding protein sequence using the genetic code. The mapping in Translation will be denoted by η_c where the subscript c denotes the number of mRNA features that define a codon. If three features are used like natural codons, $c = 3$; η_c can be defined as $\eta_c : R^{\ell_r} \to P^{\ell_p}$. R^{ℓ_r} and P^{ℓ_p} denote the ℓ_r and ℓ_p dimensional space of all mRNAs and proteins respectively. Note that $\ell_r = c\,\ell_p$ and for binary representation $R = P = \{0, 1\}$.

Consider the genetic code-like transformations presented in Tables 2 and 3. Note that the genetic code may be redundant. In other words, a unique protein feature value may be produced by several mRNA codons. This is also true for natural genetic code (Table 1). As a result, there exist many equivalent mRNA sequences that produce the same protein sequence. All these mRNA sequences have the same genetic fitness since they all map to the same protein sequence. So we can view the space of mRNAs grouped into different equivalence classes. We shall call this characteristic *Translation Introduced Equivalence* (TIE) and these groups of equivalent mRNAs will be called the TIE classes. Let R_p be the TIE class for the protein sequence $\mathbf{p}$. We can also define R_p in the following manner: $R_p = \{\mathbf{r}_j | \mathbf{r}_j \stackrel{\eta_c}{\to} \mathbf{p}\}$. The cardinality of the set R_p depends on the genetic code and the protein

sequence $\mathbf{p}$. Let a_0 and a_1 be the total number of codons that map to a protein feature value of 0 and 1 respectively. Let $\ell_{p,0}$ and $\ell_{p,1}$ be the number of 0-s and 1-s in $\mathbf{p}$ respectively. Then the cardinality of the TIE class is $|R_p| = a_0^{\ell_{p,0}} a_1^{\ell_{p,1}}$.

Since one feature in the protein sequence maps to c mRNA features, partitions defined in the mRNA and the protein spaces can be associated with each other. Let $\mathbf{j}$ and $\mathbf{j}'$ be partitions in the mRNA and the protein spaces respectively. We will call $\mathbf{j}'$, the *reflection* of $\mathbf{j}$ in the protein space when $\mathbf{j}'_i = 1$ if and only if $\mathbf{j}$ takes a value of 1 at the location(s) corresponding to at least one of the mRNA features associated with $\mathbf{j}'_i$. If $\mathbf{j}$ has zeroes at all the locations corresponding to $\mathbf{j}'_i$ then $\mathbf{j}'_i = 0$.

For example, the reflection of the partition $\mathbf{j} = 101000$ using a genetic code of codon size three is $\mathbf{j}' = 10$. The left three bits of $\mathbf{j}$ are associated with the leftmost bit of $\mathbf{j}'$. Since two of those three bits are set to 1, $\mathbf{j}'_0 = 1$. However, none of the rightmost three bits in $\mathbf{j}$ takes the value 1. So the corresponding $\mathbf{j}'_1 = 0$. Note that the reflection of 100000 is also 10 since $\mathbf{j}'_0 = 1$ as long as at least one of the leftmost three bits is set to 1. Similarly the reflection of 100110 under a genetic code of codon size three is 11.

Note that different mRNA partitions may have the same reflection in the protein space. If q is the number of ones in $\mathbf{j}'$ then it is the reflection of $(2^c - 1)^q$ different partitions in the mRNA space. The number of 1-s in $\mathbf{j}'$ will be called the *absolute order of partition* $\mathbf{j}$.

Once the protein sequence is constructed from the mRNA sequence, the protein folds into a three dimensional structure and its shape determines its fitness. Let us use $f : P^{\ell_p} \rightarrow \Re^+$ for denoting this fitness function that maps the protein sequence to a non-negative real-valued range. Since the protein sequences are produced from the mRNA sequences, we can also define the fitness over the domain of mRNA sequences. Let $\phi : R^{\ell_r} \rightarrow \Re^+$ be this fitness function defined over the mRNA representation. Therefore, $\phi(\mathbf{r}) = f(\mathbf{p}) = f(\eta_c(\mathbf{r}))$. Therefore, $\phi(\mathbf{r})$ can be viewed as a different representation of the genetic fitness function $f(\mathbf{p})$.

In this section we shall study the representations of $f(\mathbf{p})$ and $\phi(\mathbf{r})$. We will be particularly interested in the effect of the representation transformation η_c on the complexity of inducing the function. In other words, we would like to know if $\phi(\mathbf{r})$ has a more efficient description compared to that of $f(\mathbf{p})$. For example, if the size of the new representation is smaller by a considerable factor then its learning will be computationally easier. So it will be desirable over the original representation.

The rest of this paper will use two toy functions to illustrate the analytical observations. These functions are defined in the following. Let $\mathbf{x}$ be a boolean string of length ℓ and ones($\mathbf{x}$) returns the number of ones in $\mathbf{x}$.

1. Needle-in-a-haystack (NH) function:

$$\begin{aligned} f(\mathbf{x}, \mathbf{x}_{\text{opt}}) &= \ell \quad \text{if} \quad \mathbf{x} = \mathbf{x}_{\text{opt}}, \\ &= 0 \quad \text{otherwise}. \end{aligned} \tag{4}$$

   Where $\mathbf{x}_{\text{opt}}$ is the domain member with the maximum function value. Different NH functions can be defined using different choices for $\mathbf{x}_{\text{opt}}$.

2. Trap function:

$$f(\mathbf{x}) = \ell \quad \text{if ones}(\mathbf{x}) = \ell$$
$$= \ell - \text{ones}(\mathbf{x}) - 1 \quad \text{otherwise}$$

The following section explores the change in the properties of the Fourier coefficients under the genetic code-like representation transformations.

5.2 Exponential decay of individual Fourier coefficients

The j-th Fourier coefficient in the mRNA space can be defined as,

$$w_{\mathbf{j}} = \frac{1}{2^{\ell_r}} \sum_{\mathbf{r}} \phi(\mathbf{r}) \psi_{\mathbf{j}}(\mathbf{r})$$
$$= \frac{1}{2^{c\,\ell_p}} \sum_{\mathbf{p}} f(\mathbf{p}) \sum_{\mathbf{r_i} \in R_p} \psi_{\mathbf{j}}(\mathbf{r_i}) \tag{5}$$

The magnitude of the second summation in the above expression may take a value in between 0 and $a_0^{\ell_{p,0}} a_1^{\ell_{p,1}}$ (cardinality of R_p) depending upon the nature of the set R_p. This imposes a scaling factor to the contribution of every unique protein sequence to the j-th FC. Let us explore the effect of such scaling on the magnitude of an FC.

As noted earlier, the value of $\psi_{\mathbf{j}}(\mathbf{r})$ depends only on those features of $\mathbf{r}$ corresponding to the 1-s in the partition $\mathbf{j}$. The mRNA features corresponding to the positions with 1-s in the partition $\mathbf{j}$ may belong to the (1) same mRNA codon, (2) different codons, and (3) a combination of both. In other words they originate from the (1) the same protein feature (since one feature in the protein sequence maps to c features in the mRNA sequence) or (2) different protein features or (3) a combination of both respectively. Next, we are going to represent $\mathbf{j}$ using a collection of partitions $\{\mathbf{j}_0, \mathbf{j}_1, \cdots \mathbf{j}_q\}$ where $\mathbf{j}_0$ represents the null partition with all 0-s and every $\mathbf{j}_{i\neq 0}$ represents a sub-partition of the 1-contributing positions of $\mathbf{j}$ that contains only those features that belong to the same protein feature. Note that the reflection of any $\mathbf{j}_{i\neq 0}$ in the protein space has only one 1. The null partition always contribute a value of 1 and it is introduced only for taking care of the case when the partition $\mathbf{j}$ is a sequence of all 0-s. For example, consider a two-bit protein space that maps to a six-bit mRNA space. The partition 110001 in the mRNA space can be represented in terms of the sub-partitions 000000, 110000, and 000001. Note that $\psi_{110001}(\mathbf{r}) = \psi_{000000}(\mathbf{r})\psi_{110000}(\mathbf{r})\psi_{000001}(\mathbf{r})$. We can write, $\psi_{\mathbf{j}}(\mathbf{r}) = \prod_{\alpha=0,1,\cdots q} \psi_{\mathbf{j}_\alpha}(\mathbf{r})$. Therefore, we can rewrite Equation 5 as follows:

$$w_{\mathbf{j}} = \frac{1}{2^{c\,\ell_p}} \sum_{\mathbf{p}} f(\mathbf{p}) \sum_{\mathbf{r_i} \in R_p} \prod_{\alpha=0,1,\cdots q} \psi_{\mathbf{j}_\alpha}(\mathbf{r_i}) \tag{6}$$

All the defining bits (with partition value of 1) of some $\mathbf{j}_\alpha$ belong to only one protein feature by definition. Therefore, the value of $\prod_{\alpha=0,1,\cdots q} \psi_{\mathbf{j}_\alpha}(\mathbf{r_i})$ depends only on the portion of r_i defined by those q protein features. For any given combination of q protein feature values, we can define a subspace of mRNA sub-sequences.

If p_α is the protein feature value in a given $\mathbf{p}$ corresponding to the reflection of the α-partition in the mRNA space, then let R_{p_α} be the set of all mRNA codons that maps to p_α. Let $R_{j',p}$ be the Cartesian product of R_{p_α}-s for $\alpha = 1, 2, \cdots q$. Every member of $R_{j',p}$ has $c\,q$ mRNA features. For example consider $\mathbf{p} = 110$ and $\mathbf{j} = 110000010$. So $\mathbf{j}_0 = 000000000$, $\mathbf{j}_1 = 110000000$ and $\mathbf{j}_2 = 000000010$; $\mathbf{j}_1^{'} = 100$ and $\mathbf{j}_2^{'} = 001$; $p_1 = 1$, $p_2 = 0$, and $\mathbf{j}' = 101$. In case of code A, $R_{p_1} = \{100, 000, 001, 010\}$ and $R_{p_2} = \{111, 101, 110, 011\}$. Therefore $R_{101,110} = R_{p_1} \times R_{p_2}$.

Note that the basis function $\psi_{\mathbf{j}_\alpha}(\mathbf{r}_{j,p})$ is well defined for any $\mathbf{r}_{j,p} \in R_{j',p}$ for any α since the feature values of $\mathbf{r}_{j,p}$ are defined for every defining location of the partition $\mathbf{j}_\alpha$. This is indeed a slight abuse of the symbols since the length of $\mathbf{j}_\alpha$ and $\mathbf{r}_{j,p}$ are not same. However, we take that liberty since even if we pad $\mathbf{r}_{j,p}$ with 1-s and 0-s in order to make the lengths same, the outcome will be identical. This is because the corresponding values in $\mathbf{j}_\alpha$ are 0-s by definition.

Let $q_{p,j',0}$ and $q_{p,j',1}$ be the number of 0-s and 1-s in $\mathbf{p}$ that are covered by the fixed bits of $\mathbf{j}'$, the reflection of $\mathbf{j}$ in the protein space; $q_{p,j',0} + q_{p,j',1} = q$. In other words q is the total number of 1-s in $\mathbf{j}'$. Now note that every $\mathbf{r}_{j,p} \in R_{j',p}$ there are $a_0^{\ell_{p,0} - q_{p,j',0}} a_1^{\ell_{p,1} - q_{p,j',1}}$ number of strings in the corresponding $R_{j',p}$. So we can write from equation 6,

$$
\begin{aligned}
w_{\mathbf{j}} &= \frac{1}{2^{c\,\ell_p}} \sum_{\mathbf{p}} f(\mathbf{p}) \;\; a_0^{\ell_{p,0} - q_{p,j',0}} a_1^{\ell_{p,1} - q_{p,j',1}} \sum_{\mathbf{r}_{j,p} \in R_{j',p}} \prod_{\alpha = 0,1,\cdots q} \psi_{\mathbf{j}_\alpha}(\mathbf{r}_{j,p}) \\
&= \frac{1}{2^{c\,\ell_p}} \sum_{\mathbf{p}} f(\mathbf{p}) \;\; a_0^{\ell_{p,0} - q_{p,j',0}} a_1^{\ell_{p,1} - q_{p,j',1}} \prod_{\alpha = 0,1,\cdots q} \sum_{\mathbf{r}_{j,p} \in R_{p_\alpha}} \psi_{\mathbf{j}_\alpha}(\mathbf{r}_{j,p}) \qquad (7)
\end{aligned}
$$

Let $e_{j_\alpha,p}$ and $o_{j_\alpha,p}$ be the number of members in R_{p_α} that have an even and odd number ones respectively over the partition $\mathbf{j}_\alpha$. For example, if $\mathbf{j}_\alpha = 110000$ and $p = 10$ then $e_{110000,10} = 2$ and $o_{110000,10} = 2$ for code A shown in Table 2. Now using Equation 7 we can write,

$$
w_{\mathbf{j}} = \frac{1}{2^{c\,\ell_p}} \sum_{\mathbf{p}} \;\; f(\mathbf{p}) \;\; a_0^{\ell_{p,0} - q_{p,j',0}} a_1^{\ell_{p,1} - q_{p,j',1}} \kappa \prod_{\alpha = 0,1,\cdots q} |e_{j_\alpha,p} - o_{j_\alpha,p}| \qquad (8)
$$

Where $\kappa \in \{-1, 1\}$ and $|e_{j_\alpha,p} - o_{j_\alpha,p}|$ denotes the magnitude of $(e_{j_\alpha,p} - o_{j_\alpha,p})$ for all $\alpha \neq 0$. The value of $|e_{j_\alpha,p} - o_{j_\alpha,p}|$ can be determined directly from the genetic code. By definition, for the null partition $\alpha = 0$, we set $|e_{j_0,p} - o_{j_0,p}| = 1$ As before, this is done to take care of the case where $\mathbf{j}$ is comprised of only 0-s resulting in $q = 0$.

Now let us specialize this equation for code A. For this code $|e_{j_\alpha,p} - o_{j_\alpha,p}|$ is either 2 or 0 for all the partitions (except the partition with all zeros). If $|e_{j_\alpha,p} - o_{j_\alpha,p}|$ is equal to zero for any given α and the corresponding protein feature value in $\mathbf{p}$ then the overall contribution of $\mathbf{p}$ to $w_{\mathbf{j}}$ is zero. Also note that since $\mathbf{p}$ is a binary string, any feature in $\mathbf{p}$ can take only two values—0 and 1. Therefore, if $|e_{j_\alpha,p} - o_{j_\alpha,p}|$ is 0 for a certain feature entry in $\mathbf{p}$ (corresponding to j_α in the mRNA space) $|e_{j_\alpha,p} - o_{j_\alpha,p}|$ must be 0 for the complementary feature value of $\mathbf{p}$ at the same location. As a result, the corresponding coefficient $w_{\mathbf{j}}$ will be zero. Therefore for all non-zero $w_{\mathbf{j}}$-s, except the coefficient w_0, the value of $|e_{j_\alpha,p} - o_{j_\alpha,p}|$ must be equal to 2; w_0 is the Fourier coefficient for the partition with all entries set to zero.

So for all non-zero coefficients except w_0 in the representation using the code A we can write,

$$\begin{aligned} w_{\mathbf{j}} &= \frac{1}{2^{c\,\ell_p}} \sum_{\mathbf{p}} \kappa \;\; f(\mathbf{p}) \;\; 4^{\ell_p - q} \;\; 2^q \\ &= \frac{1}{2^{\ell_p + q}} \sum_{\mathbf{p}} \kappa \;\; f(\mathbf{p}) \\ &\leq \frac{1}{2^{\ell_p + q}} \sum_{\mathbf{p}} f(\mathbf{p}) \leq \frac{w_0}{2^q} \end{aligned} \tag{9}$$

Note that $w_0 = \frac{1}{2^{\ell_p}} \sum_{\mathbf{p}} f(\mathbf{p})$. This equation shows an exponential decay in the magnitude of the coefficients as the partition index of the coefficients involve more and more defining bits. As we increase the value of q (the number of 1-s in $\mathbf{j}'$, the reflection of the partition $\mathbf{j}$) the upper bound on magnitude of the coefficient $w_{\mathbf{j}}$ decreases exponentially.

We can also specialize Equation 8 for the genetic code B. Note that $|e_{j_\alpha,p} - o_{j_\alpha,p}| = 1$ for all α and $\mathbf{p}$. Also $a_1 = 1$ and $a_0 = 7$. Therefore,

$$w_{\mathbf{j}} = \frac{1}{2^{c\,\ell_p}} \sum_{\mathbf{p}} \;\; \kappa f(\mathbf{p}) \;\; 7^{\ell_{p,0} - q_{p,j',0}}$$

Figure 2(Top) shows the effect of codes A and B on the Fourier representation of function NH. The figure also shows the magnitude of the coefficients of the original Fourier representation without using the representation transformation where all the coefficients have the same magnitude. However, the magnitude decays exponentially with respect to the absolute order of the mRNA partitions for representations generated using code A and B. Note that the magnitude of the non-zero coefficients corresponding to partitions with same absolute order are same. Figure 2(Bottom) shows similar results for the Trap function.

Magnitudes of the individual coefficients do not tell the complete story. Construction of an efficient representation requires consideration of properties of all the coefficients together. This is particularly important for the current case since these transformations expand the domain and introduce many new partitions. Even if the magnitudes of individual coefficients decrease, increased number of coefficients (recall that the mRNA representation use more features) may result in no benefit towards reducing the description size of the overall function representation. In other words, a large number of small coefficients together may contribute significantly to the output of the function. The following section explores this issue. It points out that although code A offers little benefit from this perspective, properties of code B are quite encouraging.

5.3 Energy of the Fourier spectrum

The energy of the Fourier spectrum can be defined as,

$$E = \sum_{\mathbf{j}} w_{\mathbf{j}}^2 \tag{10}$$

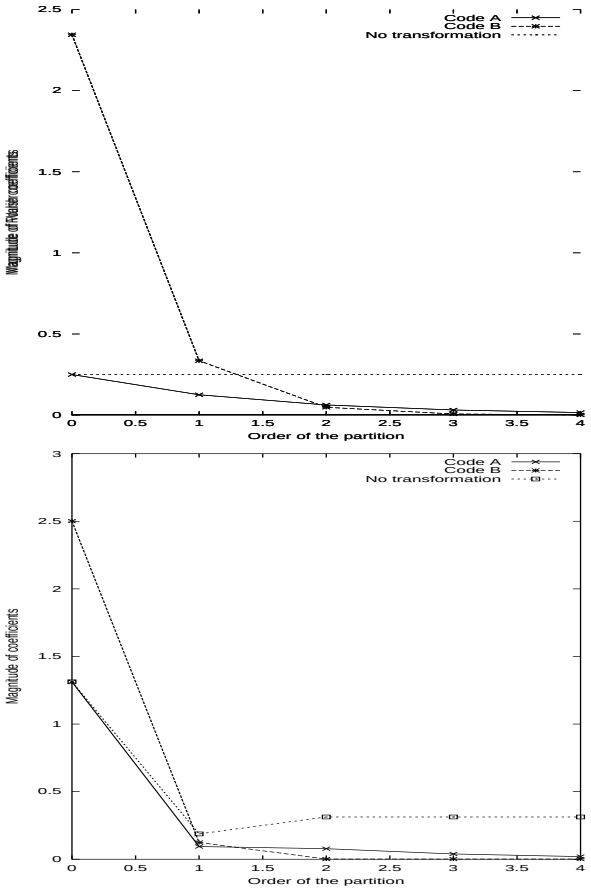

Figure 2. (Top) Variation of the magnitude of the Fourier coefficients with respect to the order (q) of the partitions in the original and transformed representations of the NH function. It shows the result using code A, code B, and no transformation. Note that the magnitude is invariant in the representation with no transformation. On the other hand it decays exponentially when the transformations are applied. (Bottom) Similar result for the Trap function. Note that all coefficients of the same order have the same magnitude for both NH and Trap functions.

Let us now study the change in the overall energy of the spectrum due to the genetic code-like representation transformations. Using Equation 5 and noting that $\psi_{\mathbf{j}}(\mathbf{x}) = \psi_{\mathbf{x}}(\mathbf{j})$ we can write,

$$w_{\mathbf{j}}^2 = \frac{1}{2^{2c\,\ell_p}} \sum_{\mathbf{p},\mathbf{s}} f(\mathbf{p}) f(\mathbf{s}) \sum_{\mathbf{r}_i \in R_p, \mathbf{r}_k \in R_s} \psi_{\mathbf{j}}(\mathbf{r}_i) \psi_{\mathbf{j}}(\mathbf{r}_k)$$

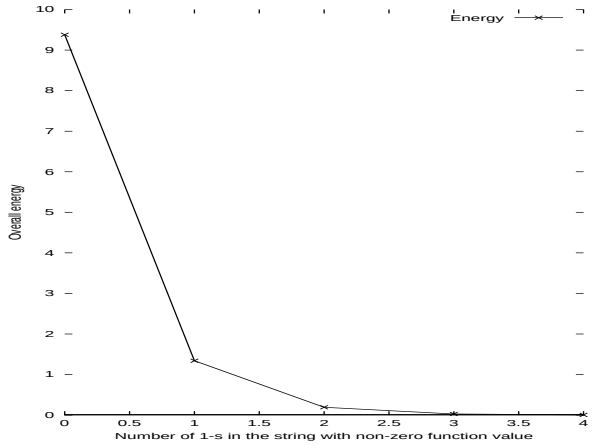

Figure 3. Variation of the overall energy with increasing number of 1-s in the string ($\mathbf{x}_{\text{opt}}$) with non-zero function value for the function NH. Code B is used.

$$\sum_{\mathbf{j}} w_{\mathbf{j}}^2 = \frac{1}{2^{2c\,\ell_p}} \sum_{\mathbf{p},\mathbf{s}} f(\mathbf{p}) f(\mathbf{s}) \sum_{\mathbf{r}_i \in R_p, \mathbf{r}_k \in R_s} \sum_{\mathbf{j}} \psi_{\mathbf{r}_i}(\mathbf{j}) \psi_{\mathbf{r}_k}(\mathbf{j})$$

Exploiting the orthonormality condition we can write,

$$\sum_{\mathbf{j}} w_{\mathbf{j}}^2 = \frac{1}{2^{c\,\ell_p}} \sum_{\mathbf{p}} f^2(\mathbf{p}) a_0^{\ell_{p,0}} a_1^{\ell_{p,1}} \tag{11}$$

Let us now specialize this result for code A. For this code, $a_0 = a_1 = 4$ and c=3. Substituting these values in Equation 11 we get,

$$E_R = \frac{1}{2^{\ell_p}} \sum_{\mathbf{p}} f^2(\mathbf{p}) = E_P$$

Where E_R and E_P are the energies of the mRNA and the protein spaces. The overall energy remains invariant under the transformation code A.

Code B however changes the overall energy. Figure 3 shows the different values of the energy for different choices of the string with non-zero function value ($\mathbf{x}_{\text{opt}}$) in the function NH using code B. The overall energy of the Fourier representation of the Trap function with four variables using code B is approximately $2.2778 E_P$.

Although the overall energy is an interesting property to observe, the most critical properties are the number of coefficients that significantly contribute to the overall energy of the representation and the location of those significant coefficients. If the number is small and the contribution from the rest is negligible, then we know that the function can be approximated using a small number of coefficients. If we also know the partitions that are associated with those significant coefficients then we should be able to efficiently compute the representation. The following section

shows that both of these requirements can be satisfied by a class of genetic code-like transformations.

5.4 Distribution of the energy in partitions of different order

The distribution of the energy among the coefficients of different order is a very interesting property of a representation. For example, if we know that a representation has a small number of significant coefficients and they are associated with a certain order of partitions then it will be easier to compute such a representation. In this section we shall study such properties of the Fourier representation produced by the genetic code-like transformations.

Recall that the order of a partition $\mathbf{j}$ is the number of ones in $\mathbf{j}$; in other words it is the number of features that define the corresponding basis function $\psi_{\mathbf{j}}(\mathbf{x})$. Let us define the *order-k energy*, $E^{(k)} = \sum_{\mathbf{j}|\text{ones}(\mathbf{j})=k} w_{\mathbf{j}}^2$. We can compute this for both the protein and the mRNA space. Note that an order-k partition in the mRNA space may correspond (through reflection) to a lower order partition in the protein space since multiple mRNA features are associated with the same protein feature. A careful study of the effect of the representation transformations on the order-k energy in the mRNA space may require understanding the properties of the coefficients in the mRNA space that correspond to exactly k features in the protein space. We are going to use the term *absolute order-k energy*, defined as $\mathcal{E}^{(k)} = \sum_{\mathbf{j}|\text{ones}(\mathbf{j}')=k} w_{\mathbf{j}}^2$; as defined earlier, $\mathbf{j}'$ is the reflection of $\mathbf{j}$ in the protein space. Just like the association between the partitions in the protein and the mRNA spaces through the concept of reflection, the distribution of energies in these two representations can be linked through the concept of absolute order-k energy.

Using Equation 7 we can write,

$$
\begin{aligned}
w_{\mathbf{j}}^2 &= \frac{1}{2^{2c\ell_p}} \sum_{\mathbf{p},\mathbf{s}} f(\mathbf{p}) f(\mathbf{s}) \;\; a_0^{\ell_{p,0}-q_{p,j',0}+\ell_{s,0}-q_{s,j',0}} a_1^{\ell_{p,1}-q_{p,j',1}+\ell_{s,1}-q_{s,j',1}} \cdot \\
&\quad \prod_{\alpha=0,\cdots q} \sum_{\mathbf{r}_{j,p}\in R_{p_\alpha}, \mathbf{t}_{j,s}\in R_{t_\alpha}} \psi_{\mathbf{j}_\alpha}(\mathbf{r}_{j,p}) \psi_{\mathbf{j}_\alpha}(\mathbf{t}_{j,s}) \\
&= \frac{1}{2^{2c\ell_p}} \sum_{\mathbf{p}} f^2(\mathbf{p}) \;\; a_0^{2(\ell_{p,0}-q_{p,j',0})} a_1^{2(\ell_{p,1}-q_{p,j',1})} \cdot \\
&\quad \prod_{\alpha=0,1,\cdots q} \sum_{\mathbf{r}_{j,p}\in R_{p_\alpha}, \mathbf{t}_{j,p}\in R_{p_\alpha}} \psi_{\mathbf{j}_\alpha}(\mathbf{r}_{j,p}) \psi_{\mathbf{j}_\alpha}(\mathbf{t}_{j,p}) + \\
&\quad \frac{1}{2^{2c\ell_p}} \sum_{\mathbf{p}\neq\mathbf{s}} f(\mathbf{p}) f(\mathbf{s}) \sum_{\mathbf{r}_i\in R_p, \mathbf{r}_k\in R_s} \psi_{\mathbf{r}_i}(\mathbf{j}) \psi_{\mathbf{r}_k}(\mathbf{j})
\end{aligned}
\tag{12}
$$

Now summing both sides of Equation 12 over all partitions and noting that the second term of the right hand side disappears because of the orthonormality property, we can write,

$$
\sum_{\mathbf{j}} w_{\mathbf{j}}^2 = \frac{1}{2^{2c\ell_p}} \sum_{\mathbf{p}} f^2(\mathbf{p}) \;\; \sum_{\mathbf{j}} a_0^{2(\ell_{p,0}-q_{p,j',0})} a_1^{2(\ell_{p,1}-q_{p,j',1})} \cdot
$$

$$\prod_{\alpha=0,1,\cdots q} \sum_{\mathbf{r}_{j,p}\in R_{p\alpha},\mathbf{t}_{j,p}\in R_{p\alpha}} \psi_{\mathbf{j}_\alpha}(\mathbf{r}_{j,p})\psi_{\mathbf{j}_\alpha}(\mathbf{t}_{j,p})$$

Comparing this with Equation 11 we note that,

$$\frac{1}{2^{c\,\ell_p}} \sum_{\mathbf{j}} a_0^{\ell_{p,0}-2q_{p,j',0}} a_1^{\ell_{p,1}-2q_{p,j',1}} \prod_{\alpha=0,1,\cdots q} \sum_{\mathbf{r}_{j,p}\in R_{p\alpha},\mathbf{t}_{j,p}\in R_{p\alpha}} \psi_{\mathbf{j}_\alpha}(\mathbf{r}_{j,p})\psi_{\mathbf{j}_\alpha}(\mathbf{t}_{j,p}) = 1 \qquad (13)$$

Now let us explore the rate of convergence of expression in the left hand side to 1. If it approaches 1 very quickly with respect to increasing order of the coefficients ($q = q_{p,j',0} + q_{p,j',1}$) then we know that only a small number low order coefficients mainly contribute to the overall energy of the Fourier representation.

Using Equations 8 and 13 we can write,

$$\frac{1}{2^{c\,\ell_p}} \sum_{\mathbf{j}} a_0^{\ell_{p,0}-2q_{p,j,0}} a_1^{\ell_{p,1}-2q_{p,j,1}} \prod_{\alpha=0,1,\cdots q} (|e_{j_\alpha,p} - o_{j_\alpha,p}|)^2 = 1 \qquad (14)$$

Although $(|e_{j_\alpha,p} - o_{j_\alpha,p}|)^2 = (e_{j_\alpha,p} - o_{j_\alpha,p})^2$ we have left the $|.|$ symbol in place since earlier we defined that $|e_{j_\alpha,p} - o_{j_\alpha,p}| = 1$ for the partition with all 0-s (i.e. $q = 0$). Equation 14 essentially controls the distribution of the eneregy with respect to the order of the partitions. GCT-s that can provide a exponential convergence of the left-hand-side of this equation to 1 with respect to increasing order, will also offer an exponential decay in the energy. .

Let us now specialize Equation 14 for code A. Since $a_0 = a_1 = 4$ and $c = 3$ for code A,

$$\frac{1}{2^{\ell_p}} \sum_{\mathbf{j}} 2^{-4q} \prod_{\alpha=0,1,\cdots q} (|e_{j_\alpha,p} - o_{j_\alpha,p}|)^2 = 1 \qquad (15)$$

This can be further simplified by counting the number of partitions of absolute order q that are associated with non-zero coefficients and noting that $|e_{j_\alpha,p} - o_{j_\alpha,p}| = 2$ for all of them. There are $\binom{\ell_p}{q}$ order-q partitions and by studying the genetic code A we observe that any protein feature corresponds to four choices (note that the null partition is not a choice) in the mRNA partition-space that can have non-zero coefficients. In other words, there are only four choices of partitions over a particular codon that can have non-zero coefficients. In a partition of absolute order equal to q there are 4^q such partitions. Therefore,

$$\frac{1}{2^{\ell_p}} \sum_{q=0}^{\ell_p} \binom{\ell_p}{q} 2^{-4q} 4^q 2^{2q} = 1$$

$$\frac{1}{2^{\ell_p}} \sum_{q=0}^{\ell_p} \binom{\ell_p}{q} = 1 \qquad (16)$$

Note that the case for null partition ($q = 0$) is taken care of since $4^0 = 1$. So the absolute order-k energy for the code A is,

$$\mathcal{E}_A^{(k)} = \frac{1}{2^{\ell_p}} \binom{\ell_p}{k} E_P; \qquad (17)$$

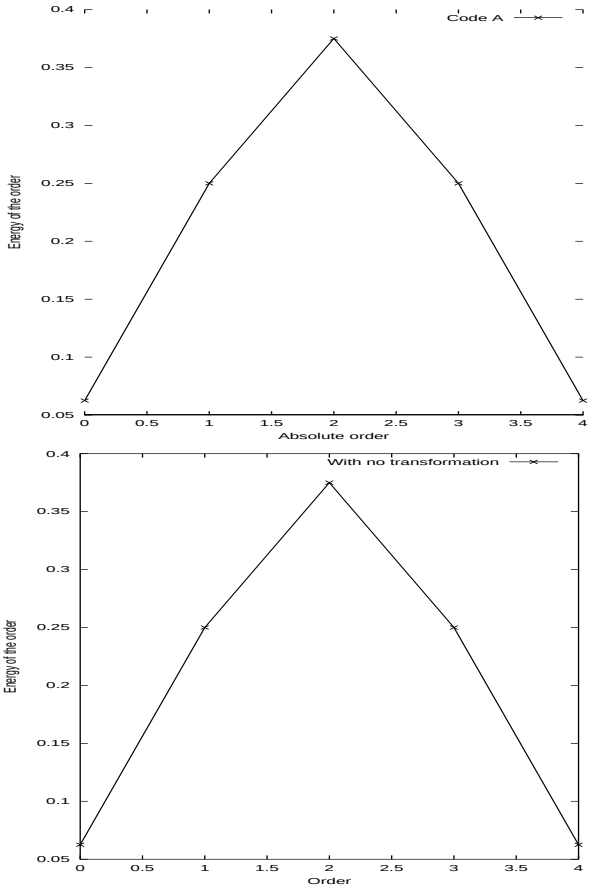

Figure 4. (Top) Distribution of the absolute order energy using code A and the function NH with $\mathbf{x}_{\text{opt}} = 0000$. (Bottom) Distribution of the energy in the protein space (i.e. with no representation transformation).

This equation clearly shows that the distribution of the energy among different orders is controlled by only the total number of partitions in the protein space with the same order. This is identical to the distribution of order-k energy in the protein space. In other words, code A does not really change the distribution of the energy among different orders. The magnitudes of the individual coefficients decay only because the order-k energy is distributed among an increased number of partitions. This also means that code A does not necessarily offer a better representation that is easier to approximate using a smaller number of coefficients.

The theoretical observations are also supported by the experimentally computed values of the Fourier coefficients. Figure 4 (Top) shows the distribution of absolute order-k energy for the function NH using code A. Figure 4 (Bottom) shows the

distribution of the energy in the protein space for the same function NH. As we see, both the distributions are identical.

Let us now specialize Equation 14 for code B. First note that $|e_{j_\alpha,p} - o_{j_\alpha,p}| = 1$ for code B. Therefore,

$$\frac{1}{2^{c\ell_p}} \sum_{\mathbf{j}} a_0^{\ell_{p,0}-2q_{p,j',0}} a_1^{\ell_{p,1}-2q_{p,j',1}} = 1$$

This can be further simplified by counting the number of partitions in the mRNA space for each absolute order value of ones($\mathbf{j}'$),

$$\frac{1}{2^{c\,\ell_p}} \sum_{q=0}^{\ell_p} \sum_{q_{p,j,0}=\max(0,q-l_p+l_{p,0})}^{\min(q,\ell_{p,0})} \binom{\ell_{p,0}}{q_{p,j',0}} \binom{\ell_p-\ell_{p,0}}{q-q_{p,j',0}} (2^c-1)^q a_0^{\ell_{p,0}-2q_{p,j',0}} = 1 \qquad (18)$$

Where q is essentially ones($\mathbf{j}'$). The term with a_1 disappeared since $a_1 = 1$ for code B. Note that the left hand side (LHS) of Equation 18 contains a summation over different values of q from zero through ℓ_p. We are interested in the convergence of the LHS to 1 as we continue to add the contributions for different values of q. In order to study that let us define,

$$g(\ell_p, \ell_{p,0}, k) = \frac{1}{2^{c\,\ell_p}} \sum_{q=0}^{k} \sum_{q_{p,j',0}=\max(0,q-l_p+l_{p,0})}^{\min(q,\ell_{p,0})} \binom{\ell_{p,0}}{q_{p,j',0}} \binom{\ell_p-\ell_{p,0}}{q-q_{p,j',0}} (2^c-1)^q a_0^{\ell_{p,0}-2q_{p,j',0}}$$

Where $0 \le k \le \ell_p$. We would like to study the convergence of $g(\ell_p, \ell_{p,0}, k)$ to 1 as k increases from 0 through ℓ_p. Note that $g(\ell_p, \ell_{p,0}, k)$ is also a function of $\ell_{p,0}$, the number of 0-s in a sequence $\mathbf{p}$. Since we are dealing with boolean sequences, $\ell_{p,0}$ is sufficient to define any particular $\mathbf{p}$.

Figure 5 shows the variation of $g(4, \ell_{p,0}, k)$ with respect to increasing k and different $\ell_{p,0}$-s. Since the convergence characteristic depends only on the number of 0-s in $\mathbf{p}$, not their exact locations in the string, the variations are shown for the four different types (note that $\ell_p = 4$) of $\mathbf{p}$-s. Figure 6 presents the variation of $g(300, \ell_{p,0}, k)$ for two boundary cases $\ell_{p,0} = 0$, $\ell_{p,0} = 300$, and the intermediate case $\ell_{p,0} = 150$.

Both Figures 5 and 6 convey an important message. Note that in both cases, $g(\ell_p, \ell_{p,0}, k)$ approaches 1 faster (with respect to k) when $\ell_{p,0}$ is large. This is simply because code B assigns seven mRNA codons for the protein feature 0. Now let us write the overall energy of the representation constructed using code B,

$$\begin{aligned} E_R &= \sum_{\mathbf{j}} w_{\mathbf{j}}^2 \\ &= \frac{1}{2^{c\ell_p}} \sum_{\mathbf{p}} f^2(\mathbf{p}) a_0^{\ell_{p,0}} g(\ell_p, \ell_{p,0}, \ell_p) \end{aligned} \qquad (19)$$

Note that the protein sequences with high genetic fitness contribute significantly to the overall energy E_R since $f^2(\mathbf{p})$ will be large for them. Moreover, the effect

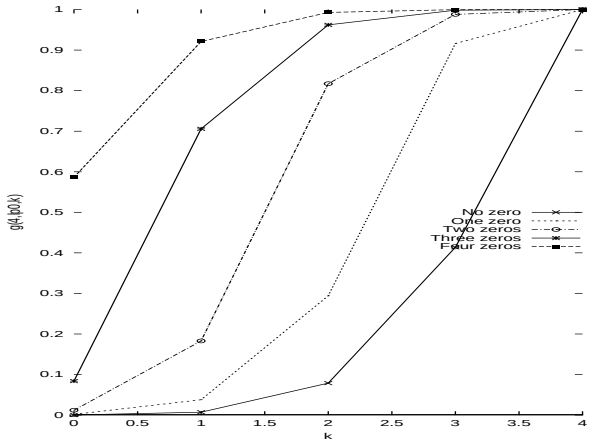

Figure 5. Variation of the $g(4, \ell_{p,0}, k)$ with respect to increasing k. The variation is shown for different types of **p**-s. $\ell_p = 4$ and the code B is used.

of $f^2(\mathbf{p})$ on the energy gets scaled up by the factor $a_0^{\ell_{p,0}}$. This essentially means that fitter protein sequences with large number of 0-s will mainly contribute to E_R. Now note that for proteins with large number of 0-s (i.e. relatively large $\ell_{p,0}$) the function $g(\ell_p, \ell_{p,0}, k)$ approaches 1 very fast. In other words, the main portion of the overall energy comes from the highly fit proteins that have more number of equivalent mRNA representations (implied by large value of $\ell_{p,0}$ and bias of code B towards the protein feature 0).

Equation 18 can also be further specialized for the function NH. If the string with all zeroes is the optimal solution then it is the only member of the domain that contribute to the Fourier coefficients. Therefore we can eliminate the summation over all p-s by only the string with all zeroes. Noting that $\ell_{p,0} = \ell_p$, $q_{p,j,0} = q$ we can write,

$$\frac{1}{2^{c\ell_p}} \sum_{q=0}^{\ell_p} \binom{\ell_p}{q} (2^c - 1)^q a_0^{\ell_p - 2q} = 1 \tag{20}$$

The theoretical observations are also supported by the experimentally computed values of the Fourier coefficients. Figure 7 shows the distribution of absolute order-k energy for the function NH using code B. As we see the contribution to the overall energy from the coefficients of a certain order diminishes as the order increases when the optimal solution contains all 0-s. The NH function is an extreme case where everyone but one domain member has a zero function value. As a result the distribution of the absolute order-k energy depends solely on the property of the sequence $\mathbf{x}_{\text{opt}}$. If we set $\mathbf{x}_{\text{opt}}$ to sequences with less number of 0-s the decay is preceded by an increase in energy.

Now let us consider a different example using the Trap functions. Note that in

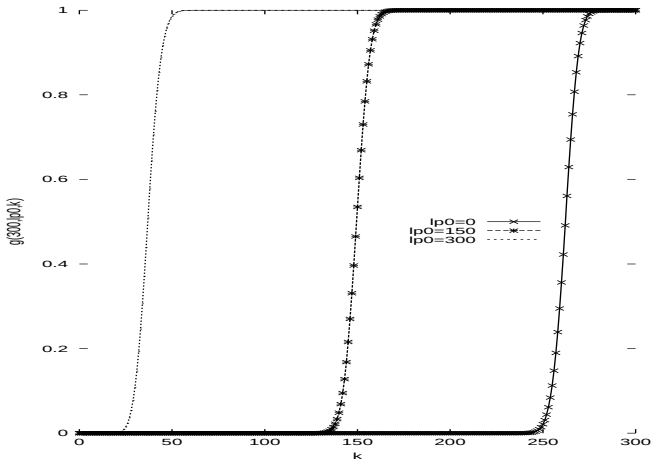

Figure 6. $g(300, \ell_{p,0}, k)$ with respect to increasing k for two boundary cases $\ell_{p,0} = 0$, $\ell_{p,0} = 300$, and the intermediate case $\ell_{p,0} = 150$. This shows that convergence rate for larger number of protein features using code B.

this case although the sequence with all 1-s has the highest function value, there are other sequences that have non-zero function value. The sequences with more number of 0-s have relatively high fitness values. This also matches with the bias of the genetic code B. Therefore we should expect a good approximation using the low order coefficients.

Figure 8(Top) shows the distribution of absolute order-k energy using the code A, code B, and no transformation for a Trap function with $\ell_p = 4$. Note that the distribution of the energy using no transformation and that using code A are identical as noted in Equation 17. Also note that the absolute order-k energy decreases exponentially for code B. Figure 8(Bottom) shows the order-k energy using both code A and B for the Trap function. Note that this is the order-k energy of the mRNA representation not the absolute order-k energy and 4-bit protein sequences map to 12-bit mRNA sequences.

It is important to realize that the match between the bias of the genetic code and the representation of the fitter proteins may not be difficult to achieve. Fitter proteins will have larger value of $f^2(\mathbf{p})$. If we assign more number of codons to the most frequent feature value (either 1 or 0 in case of binary strings) used in the fitter proteins, then the corresponding scaling factor ($a_0^{\ell_{p,0}}$ in case of code B) will also be large. For these proteins $g(\ell_p, \ell_{p,0}, k)$ also approaches 1 very fast with respect to k. In case of code B, the larger the value of $\ell_{p,0}$ in a protein, the higher the rate of convergence and the larger the scaling factor. On the other hand, the proteins with frequent feature values that have less number of codons assigned (1 in case of code B) will have a slower convergence rate for $g(\ell_p, \ell_{p,0}, k)$ and smaller scaling factor. In case of code B it will be protein sequences with smaller values for $\ell_{p,0}$

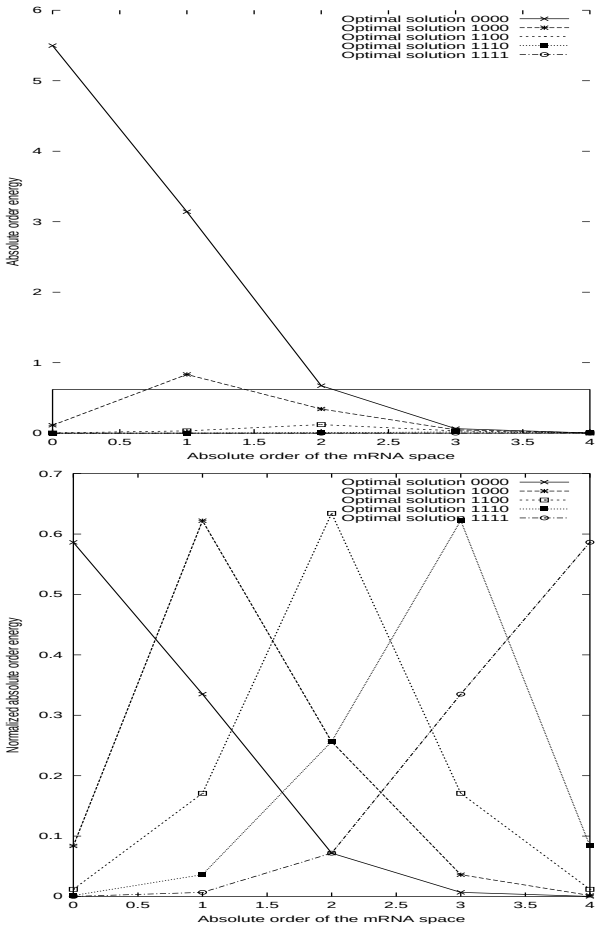

Figure 7. (Top) Distribution of the absolute order energy using Code B for function NH. (Bottom) Distribution of the normalized value of the absolute order energy using Code B for function NH.

(i.e. strings with relatively more number of 1-s). Although the convergence rate will be slow, if the fitnesses of these proteins are relatively low, their contribution to the overall energy will be low since the scaling factor will be small for them. Note that if the fitness value is relatively small compared to the scaling factor, the latter will play a more significant role. For binary representation, the issue is assigning a codon distribution among two possible protein features 0 and 1. For representations with higher cardinality, the code introduces richer transformations and we need to further explore the implications.

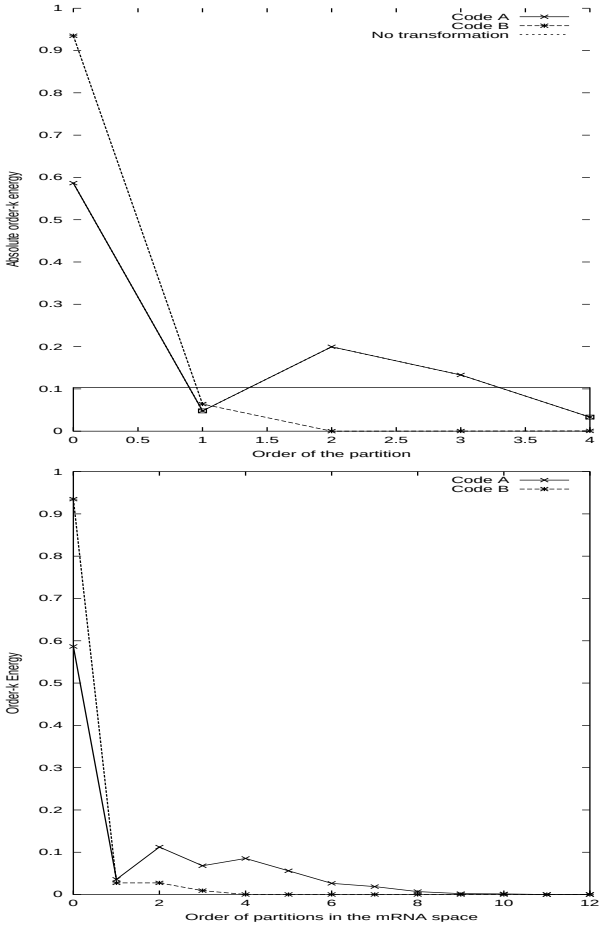

Figure 8. (Top) Distribution of the absolute order energy using Codes A and B for the Trap function. (Bottom) Distribution of the order-k energy using Codes A and B for the Trap function. Note that 4-bit protein space maps to 12-bit mRNA representation since the codon size is three.

6. Conclusions

This paper offered some intriguing properties of genetic code-like transformations that may be extremely useful for inducing a function from observed data. It showed that there exist some genetic code-like transformations that can construct a Fourier representation of some fitness functions where the low order coefficients are exponentially more significant than the higher order coefficients. This is a very critical property that allows a polynomial complexity approximation of an exponentially long function representation.

The paper demonstrated this by considering two GCT-s and a pair of functions known to have exponentially long Fourier representations. It first showed that the magnitude of individual Fourier coefficients decay at an exponential rate as the order (the number of features associated with the coefficients) increases. However, such decay in individual coefficients does not guarantee efficient representation. This is because a large number of small coefficients together may contribute an insignificant amount to the overall function value. So we needed to explore the variation of the energy (sum of the square of the coefficients) of the spectrum with respect to increasing order. We noted that one of the transformations (code B) generated an exponentially decaying energy distribution. This guarantees that a low order approximation of the function will be accurate since the cumulative contribution from the higher order terms is neglegible.

Although the results are presented in the context of specific GCT-s, this paper makes an effort to characterize the class of GCT-s that offer such useful properties. Equation 14 essentially controls this property. GCT-s that can provide a exponential convergence of the left-hand-side of this equation to 1 will also offer an exponential decay in the energy with respect to increasing order. Code B does that; however code A does not. This paper also outlines a physical conjecture for constructing such transformations. It suggests that one possible way to construct such GCT-s may be to assign more equivalent copies to protein sequences with higher genetic fitness values by introducing redundancy in the genetic code.

The implication of this paper on the field of evolutionary computation is important. A technique for efficient and scalable induction of function representation will be useful in almost every application of evolutionary algorithms. Examples include evolving programs, learning classifiers, detecting patterns from data, and optimization. This work also suggests that we should rethink our existing models of evolutionary computation. We need to further explore the computational role of gene expression. That may ultimately lead us toward unveiling the true power of genetic search.

Acknowledgments

This work was supported by the United States National Science Foundation Grants IIS-9803660 and IIS-0083946. The author would also like to thank Alden Wright, Robert Heckendorn, and Dirk Thierens for very useful comments on this paper.

References

[1] J. D. Bagley. The behavior of adaptive systems which employ genetic and correlation algorithms. *Dissertation Abstracts International*, 28(12):5106B, 1967. (University Microfilms No. 68-7556).

[2] S. Bandyopadhyay, H. Kargupta, and G. Wang. Revisiting the GEMGA: Scalable evolutionary optimization through linkage learning. In *Proceedings of the IEEE International Conference on Evolutionary Computation*, pages 603–608. IEEE Press, 1998.

[3] J. Bashford, I. Tsohantjis, and P. Jarvis. A supersymmetric model for the evolution of the genetic code. To be published in National Academy of Science USA, volume 95, issue 3, 1998.

[4] K. G. Beauchamp. *Applications of Walsh and Related Functions.* Academic Press, USA, 1984.

[5] P. Beland and T. Allen. The origin and evolution of the genetic code. *Journal of Theoretical Biology*, 170:359–365, 1994.

[6] A. D. Bethke. Comparison of genetic algorithms and gradient-based optimizers on parallel processors: Efficiency of use of processing capacity. Tech. Rep. No. 197, University of Michigan, Logic of Computers Group, Ann Arbor, 1976.

[7] A. Brindle. *Genetic Algorithms for Function Optimization.* Unpublished doctoral dissertation, University of Alberta, Edmonton, Canada, 1981.

[8] D. Dasgupta and D. R. McGregor. Designing neural networks using the structured genetic algorithm. *Artifical Neural Networks*, 2:263–268, 1992.

[9] S. Forrest and M. Mitchell. The performance of genetic algorithms on Walsh polynomials: Some anomalous results and their explanation. In R. K. Belew and L. B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 182–189. Morgan Kaufmann, San Mateo, CA, 1991.

[10] S. Fukuchi, T. Okayama, and J. Otsuka. Evolution of genetic information flow from the viewpoint of protein sequence similarity. *Journal of Theoretical Biology*, 171:179–195, 1994.

[11] D. E. Goldberg. Genetic algorithms and Walsh functions: Part I, a gentle introduction. *Complex Systems*, 3(2):129–152, 1989. (Also TCGA Report 88006).

[12] D. E. Goldberg. Genetic algorithms and Walsh functions: Part II, deception and its analysis. *Complex Systems*, 3(2):153–171, 1989. (Also TCGA Report 89001).

[13] D. E. Goldberg, B. Korb, and K. Deb. Messy genetic algorithms: Motivation, analysis, and first results. *Complex Systems*, 3(5):493–530, 1989. (Also TCGA Report 89003).

[14] R. Heckendorn and D. Whitley. Predicting epistasis from mathematical models. *Journal Of Evolutionary Computation*, 7(1):69–101, 1999.

[15] J. H. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, 1975.

[16] R. B. Hollstien. Artificial genetic adaptation in computer control systems. *Dissertation Abstracts International*, 32(3):1510B, 1971. (University Microfilms No. 71-23,773).

[17] J. Hornos and Y. Hornos. Algebraic model for the evolution of the genetic code. *Physical Review Letters*, 71(26):4401–4404, 1993.

[18] J. Jackson. *The Harmonic Sieve: A Novel Application of Fourier Analysis to Machine Learning Theory and Practice.* PhD thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 1995.

[19] F. Jacob and J. Monod. Genetic regulatory mechanisms in the synthesis of proteins. *Molecular Biology*, 3:318–356, 1961.

[20] H. Kargupta. The gene expression messy genetic algorithm. In *Proceedings of the IEEE International Conference on Evolutionary Computation*, pages 814–819. IEEE Press, 1996.

[21] H. Kargupta. Gene Expression: The missing link of evolutionary computation. In C. Poloni D. Quagliarella, J. Periaux and G. Winter, editors, *Genetic Algorithms in Engineering and Computer Science.*, page Chapter 4. John Wiley & Sons Ltd., 1997.

[22] H. Kargupta. SEARCH, computational processes in evolution, and preliminary development of the gene expression messy genetic algorithm. *Complex Systems*, 11(4):233–287, 1997.

[23] H. Kargupta. Gene expression and large scale evolutionary optimization. In *Computational Aerosciences in the 21st Century.* Kluwer Academic Publishers, 1998.

[24] H. Kargupta and S. Bandyopadhyay. Further experimentations on the scalability of the GEMGA. In *Lecture Notes in Computer Science: Parallel Problem Solving from Nature*, pages 315–324. Springer-Verlag, 1998.

[25] H. Kargupta and S. Bandyopadhyay. A perspective on the foundation and evolution of the linkage learning genetic algorithms. *Computer Methods in Applied Mechanics and Engineering*, 186(2000):269–294, 2000. Special Issue on Genetic Algorithms, Guest Editors: Goldberg, D. E. and Deb, K.

[26] H. Kargupta, D. E. Goldberg, and L. W. Wang. Extending the class of order-k delineable problems for the gene expression messy genetic algorithm. In *Proceedings of the Second Annual Conference on Genetic Programming*, pages 364–369. Morgan Kaufmann Publishers, 1997.

[27] H. Kargupta and H. Park. Fast construction of distributed and decomposed evolutionary representation. In *Late Breaking Papers of the Genetic and Evolutionary Computation Conference*, pages 139–148, 1999. Extended version is in communication.

[28] H. Kargupta and K. Sarkar. Function induction, gene expression, and evolutionary representation construction. In *Proceedings of the Genetic and Evolutionary Computation Conference, Orlando, USA. AAAI Press.*, pages 313–320. Morgan Kaufmann, 1999.

[29] H. Kargupta and B. Stafford. From DNA to protein: Transformations and their possible role in linkage learning. Proceedings of the Seventh International Conference on Genetic Algorithms, July 1997.

[30] S. Kauffman. *The Origins of Order.* Oxford University Press, New York, 1993.

[31] R. Keller and W. Banzhaf. The evolution of genetic code in genetic programming. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1077–1082. Morgan Kaufmann Publishers, 1999.

[32] S. Kushilevitz and Y. Mansour. Learning decision trees using fourier spectrum. In *Proc. 23rd Annual ACM Symp. on Theory of Computing*, pages 455–464, 1991.

[33] G. E. Liepins and M. D. Vose. Polynomials, basic sets, and deceptiveness in genetic algorithms. *Complex Systems*, 5(1):45–61, 1991.

[34] C. K. Oei. Walsh function analysis of genetic algorithms of nonbinary strings. Unpublished master's thesis, Urbana, 1992. University of Illinois at Urbana-Champaign, Department of Computer Science.

[35] S. Rana, R. B. Heckendorn, and D. Whitley. A tractable Walsh analysis of SAT and its implications for genetic algorithms. In *Proceedings of the AAAI-98*, 1998. AAAI Press.

[36] C. Reidys and S. Fraser. Evolution of random structures. Technical Report 96-11-082, Santa Fe Institute, Santa Fe, 1996.

[37] D. Rockmore, P. Kostelec, W. Hordijk, and P. Stadler. Fast Fourier transform for fitnes landscapes. Technical Report 99-10-068, Santa Fe Institute, Santa Fe, 1999.

[38] R. S. Rosenberg. Simulation of genetic populations with biochemical properties. *Dissertation Abstracts International*, 28(7):2732B, 1967. (University Microfilms No. 67-17,836).

[39] P. Schuster. The role of neutral mutations in the evolution of rna molecules. In S. Suhai, editor, *Theoretical and Computational Methods in Genome Research*, pages 287–302. Plenum Press, New York, 1997.

[40] R. E. Smith. An investigation of diploid genetic algorithms for adaptive search of nonstationary functions. TCGA Report No. 88001, University of Alabama, The Clearinghouse for Genetic Algorithms, Tuscaloosa, 1988.

[41] D. Thierens. Estimating the significant non-linearities in the genome problem-coding. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 643–648. Morgan Kaufmann Publishers, 1999.

[42] D. Thierens. Scalability problems of simple genetic algorithms. *Evolutionary Computation*, 7(4):331–352, 1999.

[43] M. Vose and A. Wright. The simple genetic algorithm and the Walsh transform: Part I, theory. *Journal of Evolutionary Computation*, 6(3):253–274, 1998.

[44] M. Vose and A. Wright. The simple genetic algorithm and the Walsh transform: Part II, the inverse. *Journal of Evolutionary Computation*, 6(3):253–274, 1998.

[45] J. L. Walsh. A closed set of orthogonal functions. *Ann. Journ. Math.*, 55, 1923.

[46] A. Wu and R. Lindsay. Empirical studies of the genetic algorithm with non-coding segments. *Journal of Evolutionary Computation*, 3(2):121–147, 1995.

[47] A. Wu and R. Lindsay. A survey of intron research in genetics. In H. Voigt, W. Ebeling, I. rechenberg, and H. Schwefel, editors, *Parallel Problem Solving from Nature-PPSN IV*, pages 101–110. Springer-Verlag, Berlin, 1996.