Computational Analysis of the Origin of the Genetic Code

A Thesis submitted in partial fulfillment of the requirements for the award of the degree of

Master of Technology
In
Computational and Systems Biology

Submitted by:
Abhay Pratap Singh
E.No: 08/75/MT/01

Under the Guidance of
Dr. Supratim Sengupta
Associate Professor

Centre for Computational Biology and Bioinformatics
School of Information Technology
Jawaharlal Nehru University
New Delhi-110067

जवाहरलाल नेहरू विश्वविद्यालय
JAWAHARLAL NEHRU UNIVERSITY
सूचना और प्रौद्योगिकी संस्थान
School of Information Technology
नई दिल्ली-110067
New Delhi-110067

Hall No.6, Lecture Hall complex JNU
Tel: (Direct) 26741517, (Off) 26704082, 26704171
Fax: 26741586, E-mail:dean_sit@mail.jnu.ac.in

CERTIFICATE

This is to certify that this M.Tech. thesis entitled "**Computational Analysis of the Origin of the Genetic Code**" has been carried out in the **Centre for Computational Biology and Bioinformatics, School of Information Technology, Jawaharlal Nehru University, New Delhi** by **Mr. Abhay Pratap Singh** under my guidance and supervision. This work is original and has not been submitted elsewhere for award of any degree or diploma.

Dr. Supratim Sengupta
Supervisor
School of Information Technology
Jawaharlal Nehru University
New Delhi

Prof. Indira Ghosh
Dean
School of Information Technology
Jawaharlal Nehru University
New Delhi

ACKNOWLEDGEMENT

I express a deep sense of gratitude towards my supervisor Dr. Supratim Sengupta for providing excellent guidance, motivation and encouragement throughout the project work which enabled me to develop a good understanding of the subject. He introduced me to this area of work and has always been with me in any case of need from initial to final stage of the work. Without his constant support and motivation, this research work could not have been accomplished successfully.

I gratefully thank our dean Prof. Indira Ghosh for providing all the resources required to carry out this research work.

I express my sincere gratitude to my teachers Prof. Indira Ghosh, Dr. N. Subbarao, Prof. R. Ramaswamy, Dr. A. Krishnamachari, Dr. Pradipta Bandyopadhyay, Dr. N.S. Sahni, Dr. Andrew Lynn, Dr. Devapriya Choudhury, and Dr. Lovekesh Vig for providing their valuable teaching throughout the course.

I would also like to extend my gratitude to my friends (Vikash, Vinod, Hungyo, Nidhi, Rama, Raja Sudhakar.D etc.) and research scholars (Ashutosh and Payal) who spent their valuable time for supporting me to finish this work.

I cannot frame the words to express my sense of gratitude to my parents for being a constant source of true love and affection, encouragement and financial help without which I could have not reached this level in my life.

I acknowledge the financial support and resources provided by D.B.T. and JNU.

Finally, I would like to thank everyone who contributed to this successful completion of the work and express my apology to those whose names I could not mention.

Abhay Pratap Singh

CONTENTS

APPENDIX

Chapter 1
INTRODUCTION

Each of the basic structures of life is eventually interpreted in terms of its evolution. If a biological system arose in some specific form as a result of evolution, it is important to understand the evolutionary pathways by which the biological system acquired its specific form.

The Genetic Code is one of the basic structures of life that might have evolved under selection to finally acquire its canonical structure. How the genetic code came into the present form (origin of the genetic code) and how it subsequently evolved from its canonical structure in certain groups of organisms have always been challenging questions to understand. In order to explain the origin and evolution of the genetic code, an attempt has to be made to predict something about an incident occurred 3.8 billion years ago on the basis of the data available at the present. Even though a lot of work has been done in this area and various theories have been proposed, many open questions about the origin and evolution of the code still remain to be answered. In this thesis we will explore the early evolution of the genetic code structure using a population genetic model of finite population size. Our aim is to explore the effect of stochasticity arising due to finite population size on the early evolution of the code structure.

The synthesis of proteins inside the cells (Figure 1.1) is governed by the process of translation that converts mRNA sequence into sequence of amino acids (protein) using the rules of genetic code. The various components taking part in the process of protein synthesis includes ribosomes, amino acids, tRNAs and aminoacyl-tRNA synthetase.

(1): Ribosomes: The translation process occurs at the ribosomes. Each ribosome consists of two subunits- a large subunit (dome-shaped) and a small subunit (oblate ellipsoid). These subunits contain ribosomal RNAs (rRNAs) and many different proteins. The large subunit contains the site of peptide bond formation which is called as peptidyl transferase center. The small subunit has a center where the information contained in mRNA sequences is decoded, and it is therefore called the decoding center. The two subunits of the ribosomes come together only at the instant when protein formation occurs and Mg^{+2} is required for this phenomenon (called association). As soon as the protein synthesis is finished, an initiation factor (IF 3 in the case of prokaryotes and eIF 2 in the case of eukaryotes) gets attached to the smaller subunit of the ribosome and the two subunits separate from each other (called dissociation).

Approximately 30 bases of mRNA are bound to the ribosome at a time. The ribosome has three tRNA binding sites called A site, P site and E site.

(2): Amino Acids: In a cell, a number of different types of proteins are synthesized. Each protein is typically a sequence of 20 different amino acids (called residues) connected to each other by peptide bonds.

(3): mRNA: mRNA is a sequence of bases (A, U, G and C) which is synthesized from DNA through the process of transcription. The mRNA is eventually converted into a sequence of amino acids through the process of translation.

(4): tRNA: tRNA (transfer RNA) is a small RNA molecule (usually about 74-95 bases) having the basic function of transferring a specific amino acid to the growing polypeptide chain at the ribosomal site of protein synthesis during the process of translation. Each tRNAs have a specific area exposed to come in contact with ribosome and the enzyme aminoacyl -tRNA synthetase.

(5): Aminoacyl-tRNA synthetase: It is an enzyme (called also as amino acid activating enzyme) which is basically required for attaching an amino acid to its specific tRNA molecule. This process of attaching an amino acid to a particular tRNA molecule is known as charging of the tRNA molecule.

The mechanism of protein synthesis (Figure 1.1) mainly involves the following steps: activation and charging of tRNAs by the appropriate amino acids, initiation, elongation and termination. Each step requires specific protein factors.

An amino acid combines with a specific aminoacyl-tRNA synthase to form a complex called aminoacyl adenylate enzyme. This process is known as activation of the amino acid. This complex binds to the 3' end of its specific tRNA molecule to form aminoacyl-tRNA molecule (charged tRNA molecule).

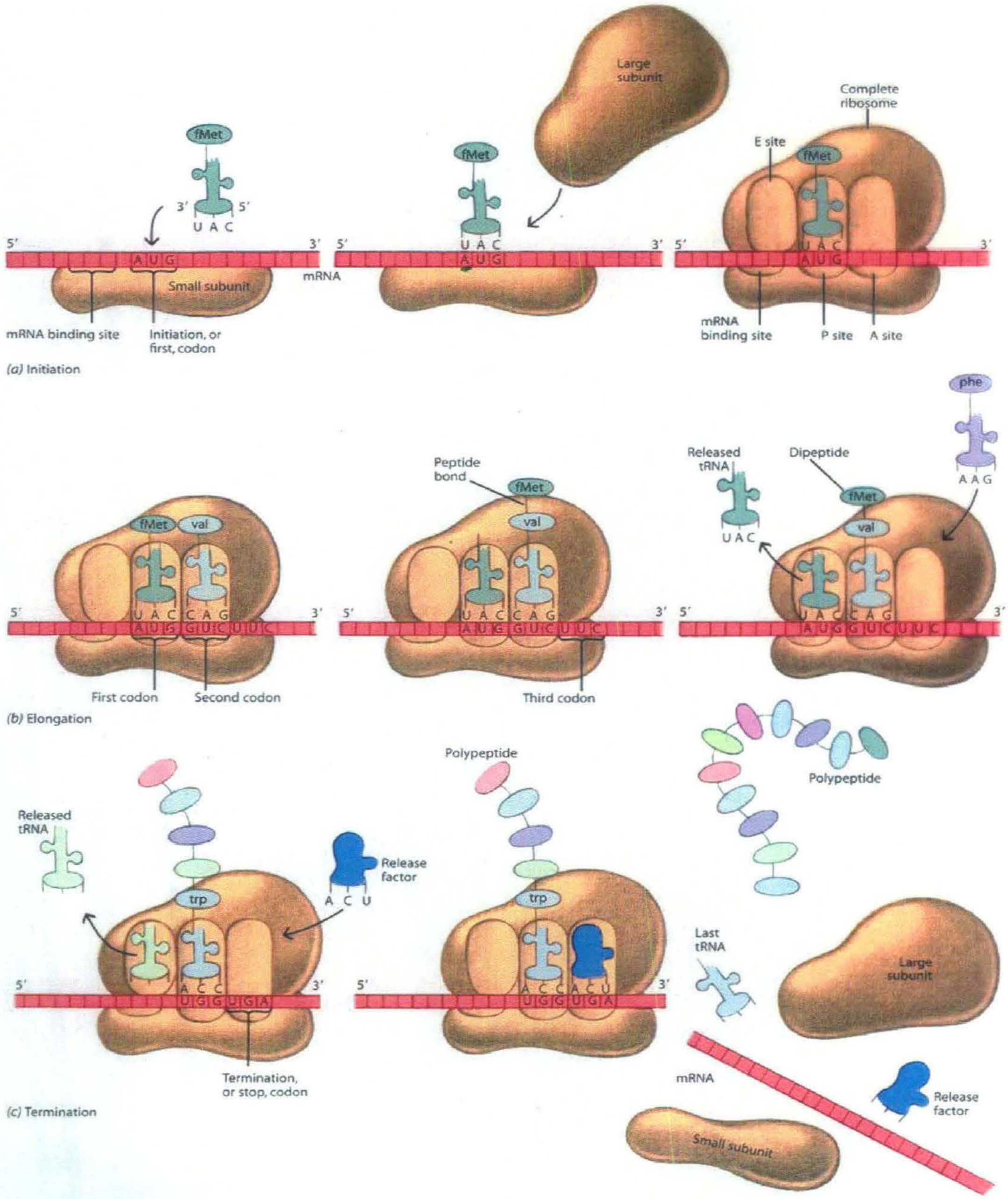


Figure 1.1

The process of translation

(Source:http://www.cps.ci.cambridge.ma.us/CRLS/LC_R/classrooms/AUGUSTINE/Gene_Expression/index_files/slide0057_image056.jpg)

The mRNA is attached to the smaller subunit of the ribosome in such a way that start codon AUG (the codon AUG codes for methionine and usually stands for initiation codon in major cases) lie at the P-site of the ribosome. A specific aminoacyl-tRNA molecule corresponding to the start codon (i.e the tRNA molecule charged with methionine: Met-tRNA) comes towards P site and its anticodon makes an unstable hydrogen bonding with the start codon of mRNA at the P site. Then the larger subunit of the ribosome arrives to combine with the smaller one. The A site of the ribosome is still exposed. A specific aminoacyl-tRNA molecule (whose anticodon is able to pair with the start codon at the site-A) reach there and pair with the next codon of the mRNA that is adjacent to the start codon. Formation of a peptide bond occurs between the carboxyl group of the amino acid attached to the tRNA molecule at the site-P and the amino group of the amino acid attached to the tRNA molecule at the site-A with the help of an enzyme called peptidyl transferase present at the larger subunit of the ribosome. Subsequently, the amino acid and tRNA molecule at the P-site are disassociated. The free tRNA molecule shifts towards the E-site of the ribosome and eventually dissociates from it. Now the site-A carries a peptidyl tRNA complex. Immediately after this process, the ribosome or the mRNA rotates (translocation) slightly in such a way that the codon at A site with peptidyl–tRNA complex reaches to the P site and a new codon next to the codon at the P site is exposed at the site-A to be attached with a specific aminoacyl-tRNA complex and the same process is repeated (elongation) until the synthesis of the polypeptide terminates when a nonsense codon (UAA, UAG and UGA) of mRNA appears at the site-A. These codons are not recognized by any of the tRNAs. As soon as a nonsense codon appears at the site-A, elongation of the polypeptide chain is terminated and is recognized by some release factors which hydrolyze the peptidyl-tRNA bond and finally the polypeptide is released. As soon as the polypeptide chain is released, the two subunits of the ribosome separate from each other.

The process of protein synthesis is error prone as during the process of translation a codon i can be mistaken for another codon j. Due to this misreading of codon i as codon j, the amino acid coded by codon i will be replaced by another amino acid corresponding to the codon j. This replacement of amino acid can have an adverse effect on the functionality and stability of the coded protein which can destabilize the protein structure and reduce its efficiency in performing its different functions. The distribution of amino acids among 64 codons in the canonical genetic code is such that neighboring codons code the amino acids with similar physicochemical properties and it has been experimentally verified that a codon is more sensitive to be mistaken for another codon in its neighborhood. Hence, the canonical code is well known to be efficient (Gilis Et al., 2010) in limiting the deleterious effects of mistranslation errors on the three dimensional structure and stability of the coded protein. The evolutionary pressure to struct the code in this particular form might be the increment in the efficiency of the code in reducing the deleterious effects of the mistranslation errors.

As fewer amino acids were thought to be codified at the early stage of the code evolution (Higgs, 2009; Wong et al. 1979, Weber et al. 1981), Higgs (2009) has recently suggested a function to calculate the average cost of a code with less than 20 amino acids. He argued that the evolutionary pressure for the addition of a new amino acid in the code is not minimization of the effect of mistranslation errors but is reduction in the cost due to the replacement of the best alternative amino acid at a site by a more suitable amino acid at that site. The new code that results from the addition of a new amino acid at the appropriate place is more efficient compared to the previous one in limiting the effect of mistranslation errors.

The evolutionary pathways for the origin of genetic code has been determined by the deterministic approach (Higgs, 2009) where the deviant code out of the many other possible alternative deviants codes arising from reassignments of some codons to a new amino acid in the population of the original codes is supposed to invade the whole population only if its cost is lower (i.e. fitness is higher) than the remaining individuals with other alternative deviant codes.

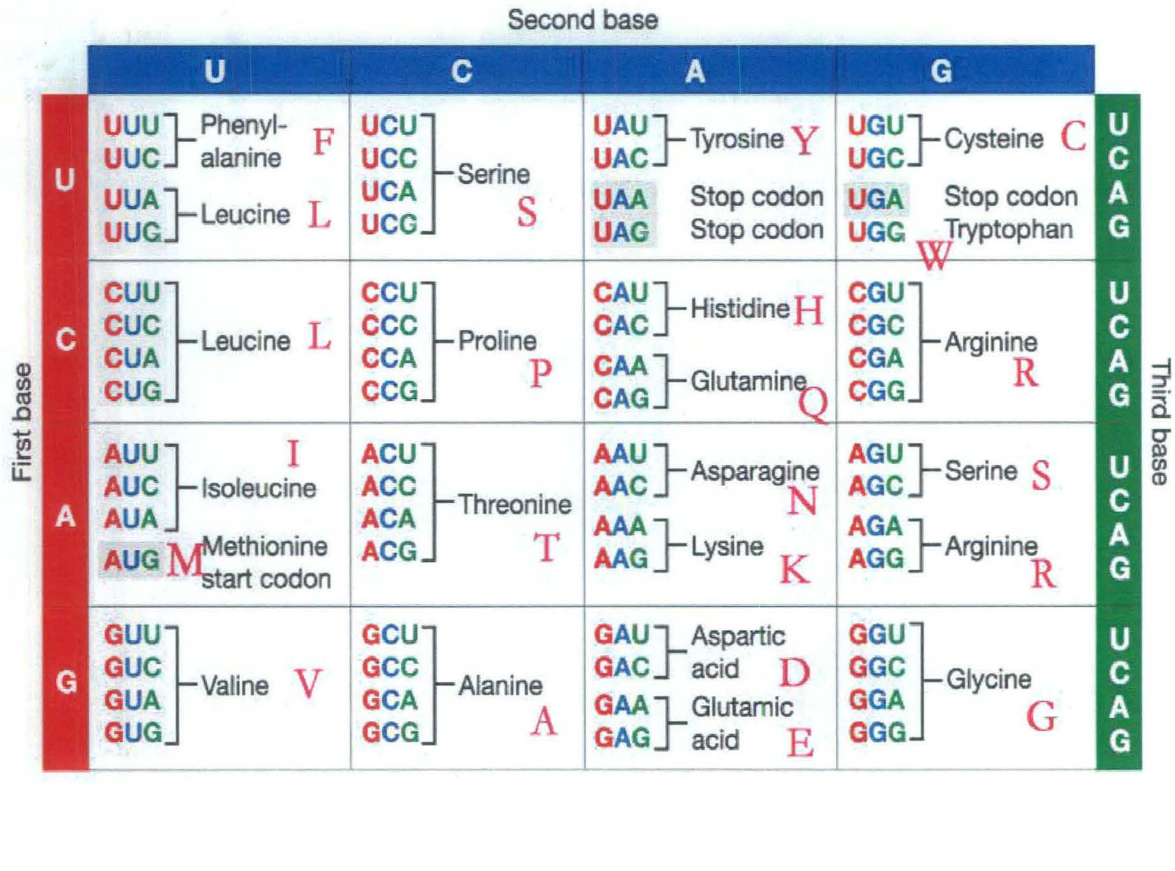
In a finite population size a alternative deviant code having cost not much higher (i.e. fitness not too much lower) than the most favored code may invade the whole population with some lower probability. We work to explore the effect of this stochasticity resulting from the finite population size on the evolution of the genetic code and we identified some cases where the pathway of the origin of the genetic code may get modified as a result of the same.

Chapter 2

BACKGROUND AND LITERATURE REVIEW

2.1 Genetic Code

The central dogma of molecular biology constitutes the following three steps; replication, transcription and translation. During the process of replication a copy of the DNA is made. Formation of mRNA occurs during the process of transcription. Finally translation is the process that converts an mRNA sequence into a sequence of amino acids that forms a protein using a specific set of rules. This set of rules constitutes the genetic code.



First base	Second base U	Second base C	Second base A	Second base G	Third base
U	UUU, UUC – Phenyl-alanine F UUA, UUG – Leucine L	UCU, UCC, UCA, UCG – Serine S	UAU, UAC – Tyrosine Y UAA Stop codon UAG Stop codon	UGU, UGC – Cysteine C UGA Stop codon UGG Tryptophan W	U C A G
C	CUU, CUC, CUA, CUG – Leucine L	CCU, CCC, CCA, CCG – Proline P	CAU, CAC – Histidine H CAA, CAG – Glutamine Q	CGU, CGC, CGA, CGG – Arginine R	U C A G
A	AUU, AUC, AUA – Isoleucine I AUG M Methionine start codon	ACU, ACC, ACA, ACG – Threonine T	AAU, AAC – Asparagine N AAA, AAG – Lysine K	AGU, AGC – Serine S AGA, AGG – Arginine R	U C A G
G	GUU, GUC, GUA, GUG – Valine V	GCU, GCC, GCA, GCG – Alanine A	GAU, GAC – Aspartic acid D GAA, GAG – Glutamic acid E	GGU, GGC, GGA, GGG – Glycine G	U C A G

Figure 2.1: Canonical genetic code

***Source**: http://campus.queens.edu/faculty/jannr/Genetics/images/codon.jpg*

Thus genetic code is a set of rules by which information encoded in mRNA sequence is translated into proteins, after DNA has beȩn transcribed into mRNA sequence.

2.2 Basic Features of Genetic Code

(1): Out of 64 codons AUG stands for start codon because process of translation is initiated by this codon which is read as amino acid methionine.

(2): 3 codons namely UUA, UAG and UGA are called as stop (nonsense) codons which terminate the synthesis of protein.

(3): More than one codon can code for a single amino acid. These are called as synonyms codons. There are 3 cases (Arg, Leu, Ser) when a single amino acid is coded by 6 codons, 5 cases (Val, Pro, Thr, Ala, Gly) when a single amino acid is coded by 4 codons, 9 cases (Phe, Tyr, His, Gln, Asn, Lys, Asp, Glu, Cys) when a single amino acid is coded by 2 codons, one case (Ile) when a single amino acid is coded by 3 codons. Thus codons in this structure are degenerate which results in a code with a block structure.

Exceptions are AUG and UGG which code only for methionine and tryptophan respectively.

(4): It is evident from the code structure that a single codon cannot code for more than one amino acid which implies that there is no ambiguity in the canonical code.

Thus we can say that the canonical code is highly degenerate but not ambiguous.

2.3 Optimality of the Genetic Code

The process of protein synthesis is error prone. During translation, a codon i can be misread as another codon j which leads to the replacement of amino acid a_i (amino acid coded by the codon i) by the new amino acid a_j (amino acid coded by the codon j) in the protein. This replacement of amino acid can have an adverse effect on the functionality and stability of the protein. Such errors can destabilize the protein structure and reduce its efficiency in performing its different functions.

The canonical genetic code has been structured in such a way that it has been found to be efficient in limiting the deleterious effects of mistranslation errors. Hence the canonical code is said to be optimized. The extent to which the canonical code has been optimized has been understood by calculating the average cost function of the canonical code and comparing it with the average cost function of other randomly generated codes which have the same block structure as the canonical code.

2.3.1 Average Cost Function

This function measures the average of cost when an amino acid a_i is substituted by another amino acid a_j due to misreading of codon i as codon j over all codons i and all misreading errors from codons i to j. It reflects the efficiency of the genetic code in limiting the effect of mistranslation errors.

The average cost function formulated by Freeland and Hurst, 1998 is given by:

$$\Phi^{FH} = (1/64)\, \Sigma_i\, \Sigma_j\, p_{ij}\, g(a_i,a_j) \qquad (1)$$

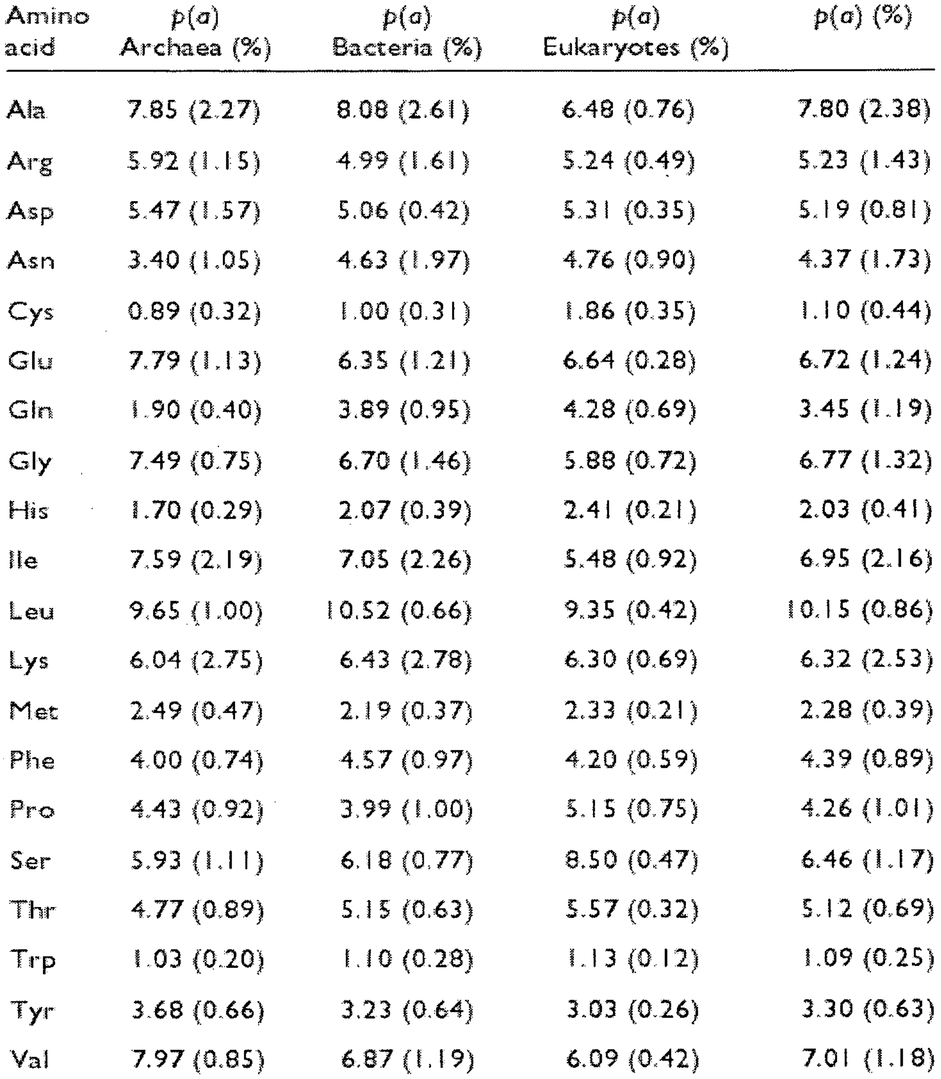
In the above cost function:

p_{ij} is the probability that a codon i will be misread as codon j.

$g(a_i,a_j)$ is the cost due to substitution of amino acid a_j in place of a_i.

1/64 stands for the frequency of codon i in genome under consideration, assuming that that each codon occurs with the same frequency in the genome.

The mean frequency of an amino acid has been calculated by averaging its frequency over genomes of the three domains of life i.e. Archea, Bacteria, Eukaryotes and given in the following table.



Amino acid	$p(a)$ Archaea (%)	$p(a)$ Bacteria (%)	$p(a)$ Eukaryotes (%)	$p(a)$ (%)
Ala	7.85 (2.27)	8.08 (2.61)	6.48 (0.76)	7.80 (2.38)
Arg	5.92 (1.15)	4.99 (1.61)	5.24 (0.49)	5.23 (1.43)
Asp	5.47 (1.57)	5.06 (0.42)	5.31 (0.35)	5.19 (0.81)
Asn	3.40 (1.05)	4.63 (1.97)	4.76 (0.90)	4.37 (1.73)
Cys	0.89 (0.32)	1.00 (0.31)	1.86 (0.35)	1.10 (0.44)
Glu	7.79 (1.13)	6.35 (1.21)	6.64 (0.28)	6.72 (1.24)
Gln	1.90 (0.40)	3.89 (0.95)	4.28 (0.69)	3.45 (1.19)
Gly	7.49 (0.75)	6.70 (1.46)	5.88 (0.72)	6.77 (1.32)
His	1.70 (0.29)	2.07 (0.39)	2.41 (0.21)	2.03 (0.41)
Ile	7.59 (2.19)	7.05 (2.26)	5.48 (0.92)	6.95 (2.16)
Leu	9.65 (1.00)	10.52 (0.66)	9.35 (0.42)	10.15 (0.86)
Lys	6.04 (2.75)	6.43 (2.78)	6.30 (0.69)	6.32 (2.53)
Met	2.49 (0.47)	2.19 (0.37)	2.33 (0.21)	2.28 (0.39)
Phe	4.00 (0.74)	4.57 (0.97)	4.20 (0.59)	4.39 (0.89)
Pro	4.43 (0.92)	3.99 (1.00)	5.15 (0.75)	4.26 (1.01)
Ser	5.93 (1.11)	6.18 (0.77)	8.50 (0.47)	6.46 (1.17)
Thr	4.77 (0.89)	5.15 (0.63)	5.57 (0.32)	5.12 (0.69)
Trp	1.03 (0.20)	1.10 (0.28)	1.13 (0.12)	1.09 (0.25)
Tyr	3.68 (0.66)	3.23 (0.64)	3.03 (0.26)	3.30 (0.63)
Val	7.97 (0.85)	6.87 (1.19)	6.09 (0.42)	7.01 (1.18)

Table 2.1

The mean frequency of amino acids

(Gilis et al., 2001)

A strong correlation between the relative frequency of an amino acid and number of synonyms codons coding for it was first observed by King et al (1969). The plot between the number of synonyms codons n (a) corresponding to an amino acid a and its relative frequency p(a) is as follows:

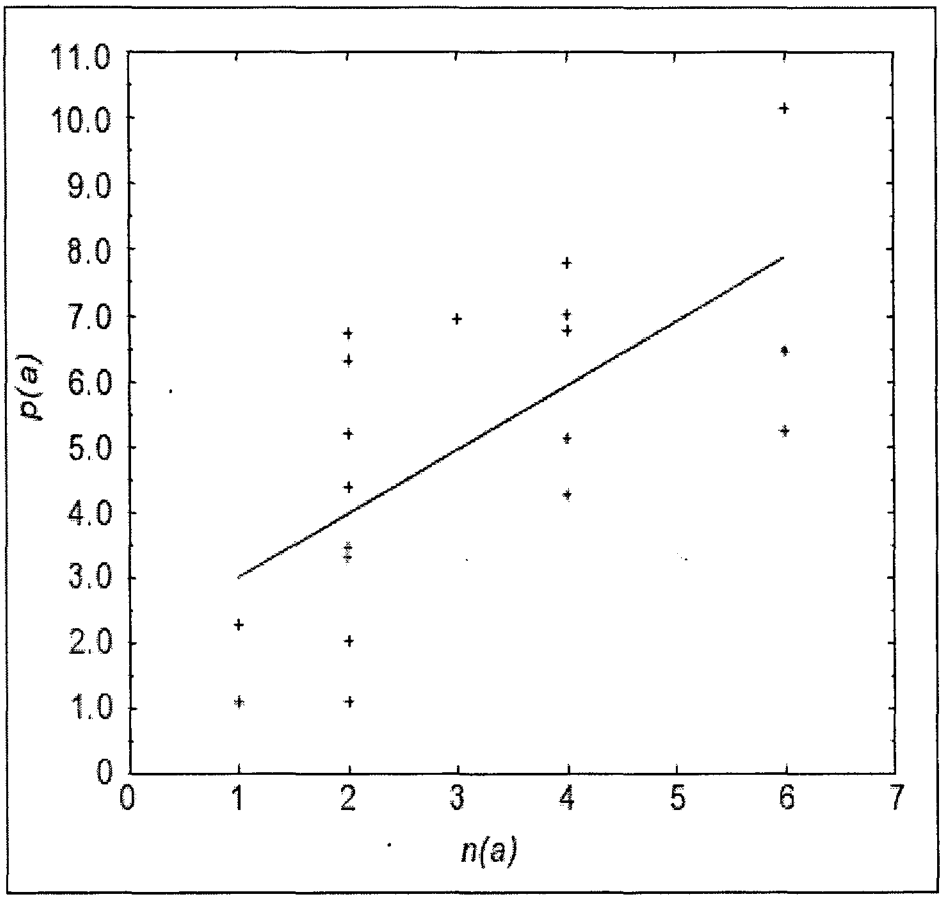


Figure 2.2

(Gilis et al., 2001)

The correlation coefficient between n(a) and p(a) was found to be 0.66.

As different amino acids occur with different frequencies (table 2.1), the correlation between n(a) and p(a) indicates that codons in genome do not occur with equal frequencies and frequency of each of the synonyms codons can be calculated from the mean frequency of the protein coded by them. Thus the correlation between the number of synonyms codons and the mean frequency of the amino acids coded by them indicates that the amino acid frequency is a important parameter to be incorporated in the average cost function.

Let $p(a_i)$ is the mean frequency of amino acid a_i coded by codon i and $n(a_i)$ is the number of synonyms codons corresponding to amino acid a_i, then frequency of each of the synonyms codons is calculated as:

$$Fi = p(a_i)/n(a_i) \qquad (2)$$

Thus average cost function formulated by Gilis et al., (2001) is as follows:

$$\Phi^{aa} = \Sigma_i \Sigma_j F_i \; p_{ij} \; g(a_i,a_j) \qquad (3)$$

Frequency of codon i, F_i is calculated with help of equation (2).

In the above cost function it has been assumed that there is no codon bias i.e. frequency of each synonyms codon corresponding to an amino acid is same.

Assuming that mean frequency of each of the amino acid is same, cost function is reduced to

$$\Phi^{equif} = (1/20)\,\Sigma_i\,(1/n(a_i))\;\Sigma_j\; p_{ij}\; g(a_i,a_j) \qquad (4)$$

The error probability matrix p_{ij} (as there are 64 codons, hence a 64/64 error probability matrix) has been calculated by taking into account the fact that transitions are more common than transversions and errors at the 3rd position of a codon is more frequent than 1st position which is again more common than errors at second position (Woes, 1965; Friedman et al., 1964; Parker, 1989).

Thus the error probability matrix is written as (Freeland and Hurst, 1998):

$p_{ij} = 1/N$ if i and j differ at 3rd position or by transition at 1st position.

$p_{ij} = 1/2N$ if i and j differ by transversion at 1st position or by transition at 2nd position.

$P_{ij} = 1/10N$ if i and j differ by transversion at 2nd position.

$P_{ij} = 0$ if i and j differ at more than one position.

Where N is a normalisation factor satisfying the equation:

$$\Sigma_j p_{ij} = 1;$$

2.3.2: Cost of Replacement of an Amino Acid by Another

The function g (a_i,a_j) gives the cost when amino acid a_i is substituted by a_j in the protein structure. Hence it should measure the difference between amino acids a_i and a_j with respect to their physicochemical properties which are responsible for the stability and functionality of the protein structure. Various cost functions have been devised to measure this cost.

Most dominating interactions for stability of protein structure are hydrophobic interactions. Hence a cost function can be defined on the basis of this single physicochemical property.

$$g^{hydro}(a,a') = (h(a) - h(a'))^2 \qquad (4)$$

h(a) and h(a') are the hydrophobicity of amino acids a and a' respectively on some scale.

The above cost function has been measured on two scales of hydrophobicity polarity and average solvent accessibility and represented by g^{pol} and g^{access} respectively. Since other properties also contribute to the stability of protein, a better cost function has been devised (Gilis et al., 2001) which measures the difference between amino acids a and a' more accurately.

$$g^{mutate}(a,a') = M(a,a') \qquad (5)$$

M (a,a') represents the change in free energy of protein when amino acid a is replaced by a'. A dataset of 141 well resolved protein structures with less than 20% sequence identity or less than 25% sequence identity and no structural similarity has been used to derive the above cost function.

M (a,a') has been calculated (Gilis et al., 2001) by averaging change in free energy when all sites occupied by a is replaced by a' over all protein structures within the dataset.

2.3.3: Canonical Code versus Random Codes

By reshuffling the amino acids in the canonical genetic code we can generate a number of random codes. By maintaining the codon block structure i.e. the codons which were synonyms in the canonical code are again synonyms in the random code, we can generate $20! = 2*10^{18}$ random codes. In order to generate random codes the stop codons are constrained to the same blocks as in the canonical genetic code.

Three cost functions Φ^{FH}, Φ^{equalf} and Φ^{faa} have been calculated (Gilis et al., 2001) corresponding to each random codes and also for the canonical code. In this way the fraction of random codes (f) having cost lower than the natural code has been estimated as in the following table.

f	Φ^{FH}	Φ^{equalf}	Φ^{faa}
g^{pol}	$9.8*10^{-7}$	$1.5*10^{-6}$	$6.5*10^{-7}$
g^{access}	$1.7*10^{-6}$	$1.9*10^{-6}$	$1.2*10^{-6}$
g^{mutate}	$2.3*10^{-6}$	$6.0*10^{-7}$	$2.0*10^{-9}$

Fraction of random codes having cost lower than the natural code

Table 2.2

It is clear from the table that the fraction f has the lower value corresponding to the function Φ^{faa} compared to other average cost functions. Hence it has been asserted that the natural code is better optimized with respect to translational errors if the frequency of amino acids is incorporated in the calculation of the average cost funcion.

As function Φ^{faa} has the smallest value if g^{mutate} is used to calculate the cost of substitution of amino acid one by another and it has been discussed that g^{mutate} measures the difference between amino acids with respect to stability of the coded protein.

Hence it can be concluded that the natural genetic code has been optimized up to a great extent in minimizing the effect of mistranslation errors on the encoded proteins.

2.4 Non Universality of the Genetic Code

The canonical genetic code was discovered around 1966 and believed to be universal (Crick, 1968) i.e. same code was being used by all organisms.

Recently it has been found that few codons have been reassigned to other amino acids in some mitochondrial and smaller nuclear genomes.

Codon reassignment is a change in the translation system such that when it occurs, the codon coding for an amino acid is now used to code for a new amino acid. At a result of this reassignment, a new amino acid is introduced at the place of the old one wherever that codon occurs in the protein-coding parts of the genome. Various mechanisms have been proposed to explain the codon reassignment process. A unified model has been proposed which describes all these mechanisms within a gain- loss framework (Sengupta et al., 2005).

Many deviations from the canonical code have been observed (Knight et al. 2001) in both mitochondrial and in a few nuclear and bacterial genomes. The evolutionary mechanisms responsible for these codon reassignments in mitochondrial genomes were analyzed by Swire et al. (Swire 2005) and more recently by Sengupta et al (Sengupta et al. 2007). Sengupta et al. were able to identify the underlying mechanism of many of the observed codon reassignments. In light of these discoveries, it is clear that the canonical genetic code is not quite universal but still evolving with time.

2.5 Origin of Genetic Code

It has been observed from the experiments performed in the prebiotic conditions that not all the 20 biological amino acids were present at the very early period of earth's history. The concentration of different amino acids present in interstellar clouds and meteorites is found to be much consistent with the concentration of amino acids suggested by Miller-Urey experiment (Wong et al. 1979, Weber et al. 1981).

The amino acids observed in these experiments with decreasing concentration, called early amino acids, are as follows:

Gly, Ala, Asp, Glu, Val, Ser, Ile, Leu, Pro, Thr.

The other 10 biological amino acids have not been found to be present in the prebiotic conditions. It has been assumed that these amino acids can't be synthesized non-biologically. They can be synthesized only by following biochemical steps occurring inside the living organisms (Higgs, 2009). Hence it is supposed that earliest code was quite simple i.e. few amino acids (which were present in prebiotic conditions with higher concentration) were initially encoded in a primordial genetic code. When new amino acids arose, they got incorporated in the code gradually at the latter stage by subdivision of codon blocks.

To explain the origin of the canonical code is formidable because here attempt is to be made to explain an incident occurred 3.8 billion years ago with the data available at present; however various theories have been proposed in order to explain the origin of the genetic code. Some of them are briefly described below.

(1): Physicochemical theory

This theory says that the evolutionary pressure to structure the canonical code in present form is reduction in physicochemical distances between those amino acids which are codified by the codons differing at single base (Sonneborn, 1965; Woes et al. 1966).

Sonneborn in 1965 observed that the evolutionary pressure for organizing the genetic code in canonical form was minimization of the deleterious effects of mutations in coding sequence. In 1966 Woese et al. further suggested that the selective pressure was the reduction of the effect of mistranslation (mistranslation is a process in which a codon is misread as another during the process of translation from m-RNA to the synthesis of proteins) errors on the coded proteins.

(2): Coevolution Theory

The Coevolution theory, proposed by Wong in 1975, attempts to explain the origin of genetic code on the basis of those biosynthetic relationships between amino acids. This theory postulates that only a small subset of the 20 amino acids was encoded in the very early stage of the genetic code evolution. These amino acids are called as precursor amino acids and they can be synthesized abiologically from inorganic chemicals available in the environment. Gradually newer amino acids were synthesized using these precursor amino acids. They are called product amino acids because they require the precursor amino acids for their synthesis. After the product amino acids were synthesized, some of the codons in the domain of precursor amino acids were reassigned to the product amino acids (Wong, 1975, 1988).

First of all, codons coding for a precursor and its product amino acids were found to be contiguous i.e. Not differing at more than one base (Nirenberg et al. 1963). This is believed to be evidence in favor of the coevolution theory. Moreover, the works of Pelc (1965) and Dillon (1973) also pointed out that the structure of genetic code might reflect the biosynthetic relationship between amino acids.

The organization of the genetic code can be understood with the help of following map which shows how the codons in the domain of precursor amino acids are conceded to product amino acids.

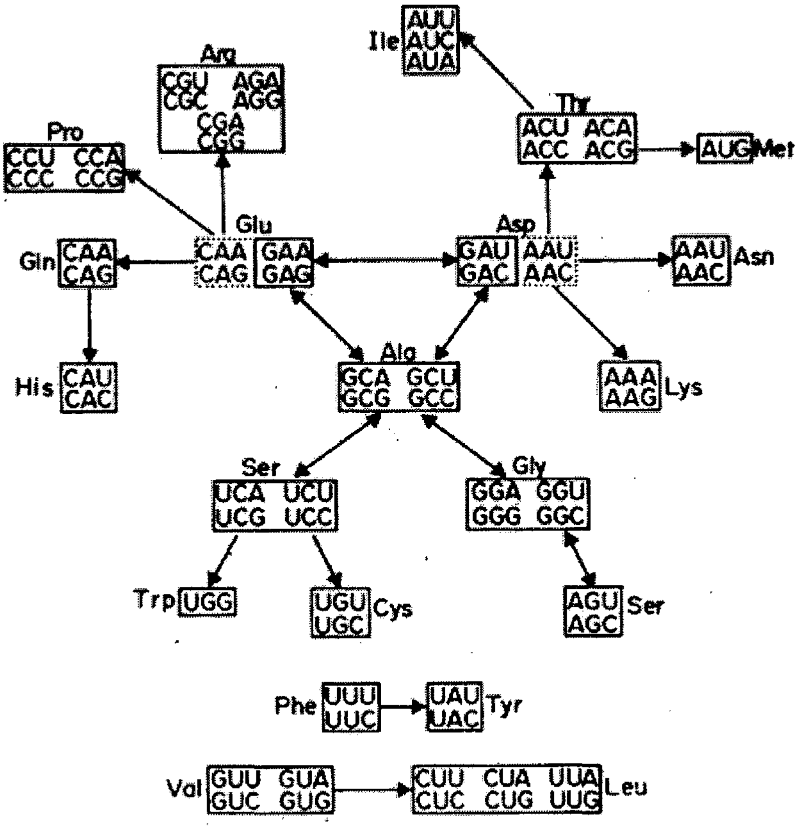


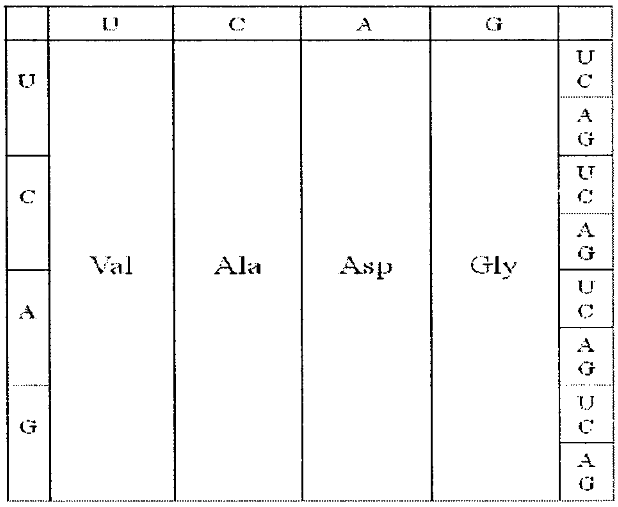
Figure 2.3

(Wong, 1975)

The figure 2.3: shows the evolutionary map of the genetic code (adopted from wong 1975).

The mapping between the codons enclosed in solid boxes and amino acids corresponding to those boxes represent the structure of the canonical genetic code. The codons enclosed within the dotted boxes in the cases of Glu and Asp is likely to be the codons which used to code these amino acids respectively at the early primordial stage. The precursor-product relationship between the amino acids is represented by the single-headed arrows where heads are directed towards the product amino acids. The double-headed arrows represent biosynthetic interconservation between the amino acids.

Very recently a four column theory has been proposed by Higgs (2009) which is basically based on the physicochemical theory which argues that the structure of the genetic code evolved to minimize the cost of mistranslation errors. As it is seen that the 5 amino acids (Gly, Ala, Asp, Glu, Val) which are found with higher concentrations in the prebiotic conditions are coded by the codons starting with G (guanine) at the first base position according to the canonical code structure. This leads to propose the earliest version of the code (Higgs, 2009) where only GNN codons used to code. This earliest stage of the genetic code has also been favored in the paper "An extension of coevolution theory of the origin of genetic code" (Di Giulio, 2008).



	U	C	A	G	
U	Val	Ala	Asp	Gly	U C
					A G
C					U C
					A G
A					U C
					A G
G					U C
					A G

Figure 2.4

Earliest version of the code (called as four column code)

Higgs, 2009

All the individual theories proposed do not explain the origin of the genetic code fully. They individually explain some aspects of the origin of genetic code. Thus we can say that the origin of genetic code can be better understood by considering all the theories in a combined way. Furthermore, to know much more about the origin of genetic code, a new theory needs to be formulated.

2.6: Cost Function for a Code with less than 20 Amino Acids

Referring to equation (1), the frequency of a codon i is calculated as

$$F_i = P(a_i)/n(a_i)$$

$p(a_i)$ is estimated from current genome sequences.

The mean frequencies of amino acids in each intermediate state, where less than 20 amino acids are encoded, are not known. Hence we can't calculate the average cost of a code with less than 20 amino acids using the previous equation (3). In order to calculate that cost a new function (Higgs, 2009) has been proposed very recently.

Suppose a gene sequence has been translated to a protein by using the canonical genetic code.

Positions occupied by a amino acid α in a protein is referred to as sites of type α. As there can be a maximum of 20 amino acids, maximum of 20 types of sites are possible at any given time. A site of type α will be preferably occupied by the same amino acid α in order to have a protein with greater stability and functionality.

If all the 20 amino acids are coded by the code then the optimal amino acid α will be used at the sites of type α. For a code with less than 20 amino acids, if the preferred amino acid α is absent from the code, sites of type α will have to be occupied by some other available amino acid and codons corresponding to that amino acid will have to be used at the sites of type α. In such cases, according to Higgs' theory, the best alternative amino acid to α will be used at sites of type α. The best alternative to α is the amino acid a_j for which g (α, a_j)=minimum among all the amino acids present in the code. Hence the codons coding for amino acid a_i will occupy those sites.

Assuming that there is no codon bias, frequency of each synonymous codon at the sites of type α can be calculated as:

$$\varphi_i(\alpha) = \delta(a_i, B(\alpha)) / n(a_i) \qquad (6)$$

$\varphi_i(\alpha) = 0$; if a_i is not the same as $B(\alpha)$ which is the best available amino acid in the absence of amino acid α

Hence the frequency of codon i in the genome can be calculated as:

$$F_i = \Sigma_i P(\alpha)\ \varphi_i(\alpha) \qquad (7)$$

Now the average cost function for a code with less than 20 amino acids is formulated as:

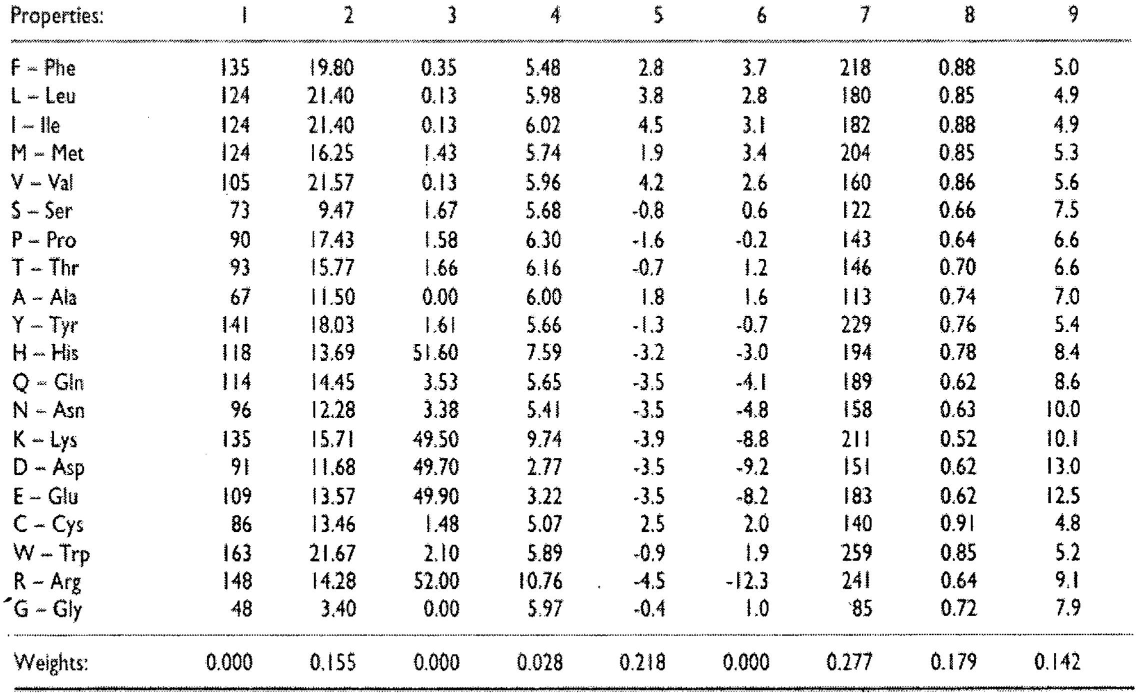
$$\Phi = \Sigma_\alpha\ \Sigma_i\ \Sigma_j\ P(\alpha)\ \varphi_i(\alpha)\ p_{ij}\ g(\alpha,a_j) \qquad (8)$$

When all 20 amino acids are available, $B(\alpha) = \alpha$ and

$$\varphi_i(\alpha) = \delta(a_{i,}\ \alpha)/n(a_i) \text{ and } \Sigma_\alpha P(\alpha)\delta(a_{i,}\ \alpha)/n(a_i) = P(a_i)/n(a_i) = F_i$$

2.6.1: A New Cost Function: g(a,b)

A new cost function, proposed by Higgs (2009), measures the difference between amino acids a and b more accurately than the previous ones by taking into account 9 different physicochemical properties given in the following table.



Properties:	1	2	3	4	5	6	7	8	9
F - Phe	135	19.80	0.35	5.48	2.8	3.7	218	0.88	5.0
L - Leu	124	21.40	0.13	5.98	3.8	2.8	180	0.85	4.9
I - Ile	124	21.40	0.13	6.02	4.5	3.1	182	0.88	4.9
M - Met	124	16.25	1.43	5.74	1.9	3.4	204	0.85	5.3
V - Val	105	21.57	0.13	5.96	4.2	2.6	160	0.86	5.6
S - Ser	73	9.47	1.67	5.68	-0.8	0.6	122	0.66	7.5
P - Pro	90	17.43	1.58	6.30	-1.6	-0.2	143	0.64	6.6
T - Thr	93	15.77	1.66	6.16	-0.7	1.2	146	0.70	6.6
A - Ala	67	11.50	0.00	6.00	1.8	1.6	113	0.74	7.0
Y - Tyr	141	18.03	1.61	5.66	-1.3	-0.7	229	0.76	5.4
H - His	118	13.69	51.60	7.59	-3.2	-3.0	194	0.78	8.4
Q - Gln	114	14.45	3.53	5.65	-3.5	-4.1	189	0.62	8.6
N - Asn	96	12.28	3.38	5.41	-3.5	-4.8	158	0.63	10.0
K - Lys	135	15.71	49.50	9.74	-3.9	-8.8	211	0.52	10.1
D - Asp	91	11.68	49.70	2.77	-3.5	-9.2	151	0.62	13.0
E - Glu	109	13.57	49.90	3.22	-3.5	-8.2	183	0.62	12.5
C - Cys	86	13.46	1.48	5.07	2.5	2.0	140	0.91	4.8
W - Trp	163	21.67	2.10	5.89	-0.9	1.9	259	0.85	5.2
R - Arg	148	14.28	52.00	10.76	-4.5	-12.3	241	0.64	9.1
G - Gly	48	3.40	0.00	5.97	-0.4	1.0	85	0.72	7.9
Weights:	0.000	0.155	0.000	0.028	0.218	0.000	0.277	0.179	0.142

Table 2.3

Paul G Higgs (2009)

The 9 physicochemical properties are

1: Volume from van der Walls radii (Creighton TE, 1993).

2: Bulkiness which measures the shape of the side chain (Zimmerman et al., 1968).

3: Polarity (Zimmerman et al., 1968).

4: Isoelectric point (Zimmerman et al., 1968).

5: Hydrobhobicity scale (Kyte et al., 1982).

6: Hydrobhobicity scale (Engleman et al., 1986).

7: Surface area accessible to water in an unfolded peptide (Miller et al., 1987).

8: Fraction of accessible area lost when protein folds (Rose et al., 1985).

9: polar requirement (Woese et al., 1966).

$$g(a,b) \; = \; d_W(a,b) = const. \times \left(\sum_k w_k (z_{ka} - z_{kb})^2 \right)^{1/2} ,$$

Where-

$g(a,b) = d_w(a,b)$ is the cost of replacement of amino acid a by b.

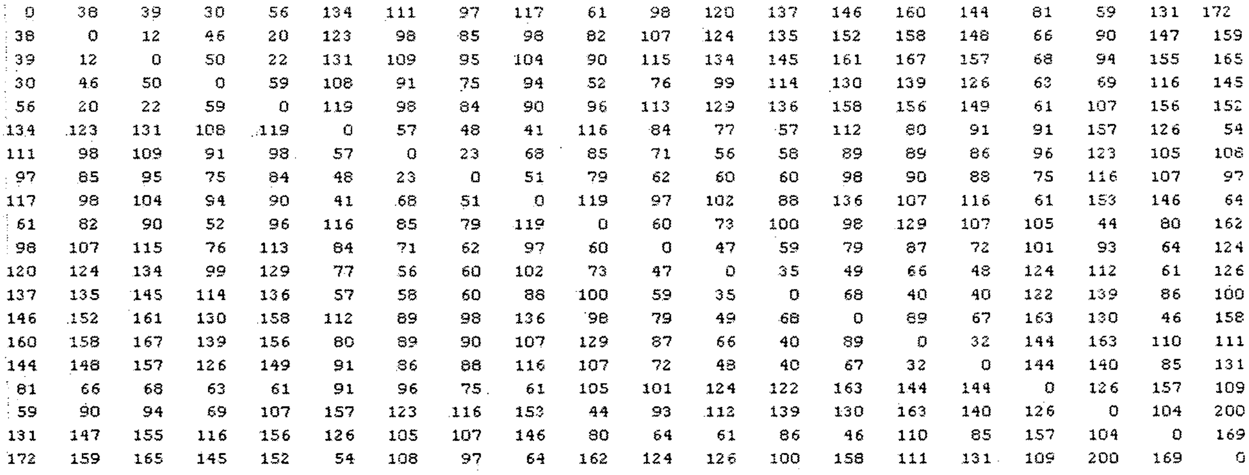
z_{ka} is the normalised kth property of amino acid a and z_{kb} is the normalised kth property of amino acid b.

w_k is the weight corresponding to that property.

An evolutionary model was fitted to sequence data with the help of maximum likelihood method and the values of the weights were determined.(Higgs et al., 2007)

Initial constant has been chosen in such a way that mean of distances between pairs of non-identical amino acids is 100.

The matrix (of size 20x20) to give the value g(a,b) has been estimated (Higgs, 2009) and is as follows:



0	38	39	30	56	134	111	97	117	61	98	120	137	146	160	144	81	59	131	172
38	0	12	46	20	123	98	85	98	82	107	124	135	152	158	148	66	90	147	159
39	12	0	50	22	131	109	95	104	90	115	134	145	161	167	157	68	94	155	165
30	46	50	0	59	108	91	75	94	52	76	99	114	130	139	126	63	69	116	145
56	20	22	59	0	119	98	84	90	96	113	129	136	158	156	149	61	107	156	152
134	123	131	108	119	0	57	48	41	116	84	77	57	112	80	91	91	157	126	54
111	98	109	91	98	57	0	23	68	85	71	56	58	89	89	86	96	123	105	108
97	85	95	75	84	48	23	0	51	79	62	60	60	98	90	88	75	116	107	97
117	98	104	94	90	41	68	51	0	119	97	102	88	136	107	116	61	153	146	64
61	82	90	52	96	116	85	79	119	0	60	73	100	98	129	107	105	44	80	162
98	107	115	76	113	84	71	62	97	60	0	47	59	79	87	72	101	93	64	124
120	124	134	99	129	77	56	60	102	73	47	0	35	49	66	48	124	112	61	126
137	135	145	114	136	57	58	60	88	100	59	35	0	68	40	40	122	139	86	100
146	152	161	130	158	112	89	98	136	98	79	49	68	0	89	67	163	130	46	158
160	158	167	139	156	80	89	90	107	129	87	66	40	89	0	32	144	163	110	111
144	148	157	126	149	91	86	88	116	107	72	48	40	67	32	0	144	140	85	131
81	66	68	63	61	91	96	75	61	105	101	124	122	163	144	144	0	126	157	109
59	90	94	69	107	157	123	116	153	44	93	112	139	130	163	140	126	0	104	200
131	147	155	116	156	126	105	107	146	80	64	61	86	46	110	85	157	104	0	169
172	159	165	145	152	54	108	97	64	162	124	126	100	158	111	131	109	200	169	0

Table 2.4

(Higgs, 2009)

2.6.2: Error Probability Matrix

The 64x64 parameter error probability matrix was reduced to a single parameter ϵ (Higgs, 2009) which controls the rate of mistranslation errors. This is similar to the p_{ij} matrix defined earlier (Freeland and Hurtz, 1998).

$p_{ij} = \epsilon$ if i and j differ at the 3rd position or by a transition at 1st position.

$p_{ij} = \epsilon/2$ if i and j differ by transversion at 1st position or a transition at second position

$p_{ij} = \epsilon/10$ if i and j differ by transversion at 2nd position.

$p_{ij} = 0$ if i and j differ at more than one position.

The probability for correct translation is calculated as:

$$p_{ii} = 1 - \Sigma_{i \neq j} p_{ij}$$

2.7 Driving Force for Addition of New Amino Acids

Substituting $\epsilon = 0$ i.e $p_{ij}=0$; in the average cost function (eqn.8):

The above cost function is reduced to the following one:

$$\Phi_0 = \Sigma_\alpha \Sigma_i P(\alpha)\, \varphi_i(\alpha)\, g(\alpha, a_i) \qquad (9)$$

Where $\varphi_i(\alpha)$ is given by eqn.6. As this cost term arises even if there is no translation error, Φ_0 is the cost of using the best available amino acid instead of the preferred amino acid at sites of type α. Since fitness generally decreases linearly with cost, the term Φ_0 can also be interpreted as reduction in fitness of coded proteins due to unavailability of preferred amino acid for a specific site type. Thus we can say that the driving force for incorporating a new amino acid in the code is reduction in the cost term Φ_0.

2.8: Change in Cost after Addition of a New Amino Acid

Suppose at any stage of the code evolution the code structure is a_i^{cur} which represents the amino acid corresponding to the codon i. The average cost is given as,

$$\Phi^{cur} = \Sigma_\alpha \Sigma_i \Sigma_j P(\alpha)\, \varphi_i^{cur}(\alpha)\, p_{ij}\, g(\alpha, a_j^{cur}) \qquad (10)$$

If a change happens in current code such that a new amino acid is incorporated in the code by reassignment of some codons, this leads to change of the code structure and new mapping between codons and amino acids with the new code structure is given by the mapping a_i^{new}. Immediately after the assignment of some codons to new amino acid, the same gene sequence has to be translated to synthesize proteins using the new code. This is because the protein coding gene sequences will not have had time to adapt to the new code and therefore they will be the same as they were just before the change in code structure. This stage before gene sequences have adapted to the new code is said to be intermediate state of the code. Thus the average cost function of the code in the intermediate stage is:

$$\Phi^{int} = \Sigma_\alpha \Sigma_i \Sigma_j P(\alpha)\, \varphi_i^{cur}(\alpha)\, p_{ij}\, g(\alpha, a_j^{new}) \qquad (11)$$

Gradually there may be many mutations in the gene sequences and eventually the codon frequencies will be equilibrated to the new code.

The average cost of the new code after its adoption:

$$\Phi^{new} = \Sigma_\alpha \Sigma_i \Sigma_j P(\alpha)\, \varphi_i^{new}(\alpha)\, p_{ij}\, g(\alpha, a_j^{new}) \qquad (12)$$

The difference of cost between the new code (after equilibration) and the old code:

$$\Delta\Phi = \Phi^{new} - \Phi^{cur}$$

The difference of cost of the new code in intermediate state (before equilibration) and the old code:

$$\delta\Phi = \Phi^{int} - \Phi^{cur}$$

The value of $\Delta\Phi$ is generally negative for addition of any new amino acid to any position in the code. Because for every site type α the best available amino acid in the code will be more closer than the previous one or it will be same as earlier, whenever a new amino acid is added to the code which leads to decrease in cost of the new code arised.

The value of $\delta\Phi$ is generally dependent on three effects arised due to addition of a new amino acid.

Suppose for sites of type a the best available amino acid in the present code is b=B(a). Here B(a) stands for a function giving the best available amino acid for the sites of type a. All codons coding for amino acid b will be used at the sites of type a.

If some of the codons coding for b are reassigned to a new amino acid a. Then

(1): The codons reassigned to amino acid a will be used at right place for sites of type a.

(2): There are other sites of type b where the codons reassigned to a are not being used at most appropriate sites but they were being used at right place before addition of new amino acid.

(3): The incorporation of new amino acid may change the cost at the other sites different from a and b.

It has been seen that the value of $\delta\Phi$ is negative only for addition of some specific amino acid to some specific positions in the code which is found to be consistent with the structure of canonical genetic code. Thus we can say that natural selection will favor the addition of a new amino acid to those positions in the code for which $\delta\Phi$ is negative.

Chapter 3
OBJECTIVES AND METHODS

Let the number of individuals using a specific genetic code at any stage of evolution be N. Due to a specific change in the translation apparatus (such as origin of a new tRNA charged with a new amino acid), a new (deviant) code arises as a result of reassignment of one or more codons to the new amino acid. The population structure will evolve as a result of selection between the two types of the individuals.

If the deviant code has fitness (1+s) relative to the original code which has fitness value of 1 (i.e deviant code is advantageous) then the probability that the new code will become fixed in the population is given by

$$p_{fix} = (1 - e^{-2s})/(1 - e^{-2Ns}) \qquad (13)$$

Where s is the selection coefficient which gives the selective advantage of the deviant code over the original code.

Case (1): If the deviant code is extremely advantageous i.e. $s >> 1$, the above eq(13) is reduced to

$p_{fix} = 1$ i.e. the deviant code is fixed with probability one.

Case (2): If the deviant code is slightly advantageous such that $s << 1$ and $N*s >> 1$ then the above eq(13) is approximately reduced to

$$p_{fix} = 2s$$

which shows the deviant code will be spread in the whole population with a small probability determined by the selection coefficient.

Case (3): For neutral selection i.e. s is equal to zero then the eq (13) reduces to:

$$p_{fix} = 1/N \qquad (14)$$

It is evident from this equation that in the case of neutral selection (all types of the individuals have the same fitness value) the fixation probability for a deviant code is inversely proportional to the population size.

If the deviant code has fitness 1- s, relative to the original code which has fitness value 1 (i.e deviant code is deleterious) then the fixation probability of the deviant code is given by

$$p_{fix} = (e^{2s} - 1) / (e^{2Ns} - 1) \qquad (15)$$

From the above equation, if population size is infinite i.e. N is very large, $p_{fix} = 0$.

Hence in a very large population size, a single deviant code having fitness lower than the remaining original codes is always outcompeted by the original (i.e. it can't be fixed).

For a finite population size (when N is not too large), if s is not too much less than one i.e. fitness of the deviant code is slightly smaller than the original code then the single deviant code can't be always outcompeted by the individuals following the original code and can get fixed with some small probability.

The evolutionary pathways for the origin of genetic code have been determined by the deterministic approach (Higgs, 2009). In this approach, if more than one deviant code arise as a result of reassignments of some codons to a new amino acid in the population which primarily consists of the original codes, the code whose cost is lowest (i.e. fitness is highest) out of all the alternative deviant codes, will spread through the whole population because this code will be favored by the natural selection.

For an infinite population size where a single deviant code having lower fitness (greater cost) than the remaining resident codes is always outcompeted with probability one. On the other hand, in a finite population size, a single deviant code having lower fitness than the remaining resident codes can get fixed with a small probability. Therefore, the fixation of alternative code structures may be affected by the population size. An alternative deviant code having cost greater (i.e. with lower fitness) than the most favored deviant code (the code with lowest cost), it may invade the whole population. The objective of this thesis is to explore the pathways of early genetic code evolution and examine how they may be constrained by the stochasticity arising as a result of finite population size.

3.1 Calculation of the Average Cost of a Code

The cost function (equation 8 discussed in section 2.6) to calculate the average cost of a code with less than 20 amino acids is as follows:

$$\Phi = \Sigma_\alpha \Sigma_i \Sigma_j P(\alpha)\, \varphi_i(\alpha)\, p_{ij}\, g(\alpha,a_j)$$

There are 4 terms used in the above cost function which are defined as follows

(a): **P(α)** : the frequency of 20 types of sites in the genome

We took this from the Table 1 given in (Higgs, 2009) which gives the average frequency of 20 amino acids in genome.

(b): p_{ij} : the probability that a codon i is misread as another codon j during the process of translation.

The 64 codons have been labeled by numbers from 1 to 64 reading them column wise in the canonical genetic code.

We constructed a matrix of size 64/64 (as there are 64 codons) considering the constraints defined in section (2.6.2) choosing the parameter value ϵ (which controls the rate of mistranslational error) as 0.05. The ith row of this matrix gives the probability of misreading the codon i as one of the other codons. The code (written in matlab) to calculate p_{ij} is given in the appendix A.

(c): **g(α,a$_j$)**: the cost due to replacement of the amino acid preferred at sites of type α by the amino acid encoded by the codon j according to the code under consideration.

We used the matrix (of size 20 by 20) discussed earlier in section (2.5.1) and given in the Table (2.3). We set $g(\alpha,a_j) = 0$, if the codon j corresponds to a stop codon.

(d): $\varphi_i(\alpha)$: the frequency of codon i at the sites of type α

α stands for sites of a particular type which is preferred by the amino acid α and its value varies according to the order or appearance of amino acids in the 1st column of Table 2; (for example: α=1 stands for sites preferred by amino acid phenylalanine (Phe), α =2 stands for sites preferred by amino acid leucine (Leu) and similarly α =20 stands for the sites preferred by the amino acid glycine (Gly). Let us suppose that we have to calculate the frequency of codons at the sites of type α = 1 at the earliest stage of the genetic code evolution i.e. the four column code discussed in section 2.5.

As the preferred amino acid for the sites of type α =1 is phenylalanine (Phe) which is not present in the four column code, the codons coding for the best available amino acid ($B(\alpha)$) in the code for the sites of type α =1 will be used at those particular sites.

Using the Table (2.4) which gives the cost of replacement of amino acid by another, we find

$g(1,5) = 56$
$g(1,9) = 117$
$g(1,15) = 160$
$g(1,20) = 172$

As g (1, 5) is minimum i.e. replacement of amino acid which is preferred at the sites of type =1 by the amino acid (5) which stands for valine (Val) has the lowest cost. Hence the best available amino acid is valine (Val) for the sites of type α =1. Therefore, codons coding for this amino acid will be used at the sites of type α =1.

n(Val) = number of synonyms codons corresponding to amino acid valine = 16

$p(\alpha=1)$ =frequency of sites of type (α=1) in the genome = p(Phe) = 0.0439.

Therefore, assuming that there is no codon bias, frequency of each of the 16 codons at the sites of type α=1 i.e sites of type Phe is calculated as

$p(\alpha=1) / n(B(\alpha)) = p(Phe)/n(Val) = (0.0439)/(16) = 0.00274375$

Thus $\varphi_i(\alpha) = 0.00274375$ if a_i i.e. amino acid corresponding to the codon i is Val.

$\varphi_i(\alpha) = 0$ if a_i i.e. amino acid corresponding to the codon i is not Val.

Since there are 64 codons distributed over twenty different site-types, we wrote a code using matlab (given in Appendix B) to construct a matrix of size 20 by 64 where each row gives the distribution of each of the 64 codons at sites of a particular type (1rst row gives distribution of codons at the sites of type $\alpha = 1$, similarly 2nd row gives the distribution of codons at the sites of type $\alpha = 2$ and so on).

In this way we calculated the average cost (matlab script for this included in Appendix B) of a code with less than 20 amino acids.

3.2: Algorithm for Selection between Individuals of the Population

(1): We start with a population of N individuals. (N-q) individuals follow the old code and the q remaining individuals follow the deviant codes which are arose in the population as a result of reassignment of some codons to other amino acids relative to those in the old code. Thus the population structure is comprised of (q+1) different type of the code.

(2): We calculate the cost (generally intermediate state cost i.e. the cost of the code just after the reassignment) of the each type of the codes present in the population using the procedure mentioned earlier.

Let us suppose that C_{old} is the cost of the old code and C_1, C_2, C_3 .... and C_q are the cost of the new (deviant) codes respectively.

We normalize the cost of the individual codes as follows:

$$\Phi_{old} = C_{old} / (C_{old}+C_1+C_2+.....+C_q)$$

$$\Phi_q = C_q / (C_{old}+C_1+C_2+.....+C_q) \; ; \; q=1,2,3.....$$

Supposing that fitness decreases linearly with cost (Higgs 2009), the fitness of each of the codes can be calculated as

$$W_{old} = 1 - s\,\Phi_{old}$$

$$W_q = 1 - s\,\Phi_q \; ; \; q = 1,2,3.....$$

Where s is a parameter which represents the strength of selection against translational errors.

(3): For each generation we choose codes randomly (one at a time) from the population and let it replicate with a probability proportional to its fitness. This process is repeated until the entire population of N individuals in the next generation has been selected. Thus population size remains fixed in every generation.

(4): Continue the above procedure until the entire population consists of only one type of code i.e. Only one of the (q+1) codes is fixed in the population.

(5): Repeat the above simulation for a number of trials (Nt) where after each trial entire population consists of only one type of code.

(6): Calculate the fraction of trials for which each type of code gets fixed in the population, to estimate the probability of fixation of each code

Let us suppose that the original code (old code) has been fixed Nt_{old} times and q different deviant codes are fixed Nt_1, Nt_2, . . . and Nt_q times respectively. Then the fixation probability of different codes can be calculated as

Fixation probability of the original code = Nt_{old} / N_t

Fixation probability of qth deviant code = Nt_q / N_t

We can also see how the structure of population changes from one generation to another, by plotting a graph showing the change in frequency of each type of code over generations.

Two codes (written using matlab) following the above algorithm, to calculate the fixation probability and to see how the structure of population varies over generations, are given in Appendix C and D respectively.

Chapter 4

RESULTS AND DISCUSSIONS

(4.1): First Column Changes

(a): The calculated cost values corresponding to different codes arised due to addition of amino acids corresponding to the codon bolock (UUN+CUN) in the four column code are listed in the following Table.

Cost of the four column code (Φ^{old}) = 42.5414

Amino Acids	**Intermediate state cost(Φ^{int})**	**Cost after adoption (Φ^{new})**	$\delta\Phi = (\Phi^{int} - \Phi^{old})$
Phe	43.7284	38.5592	1.1870
Leu	41.1029	38.5443	**−1.4385**
Ile	41.6132	39.3419	**−0.9282**
Met	44.6213	38.7678	2.0799
Val	42.5414	42.5414	0.0000
Ser	57.9470	41.2199	15.4056
Pro	53.9812	38.9357	11.4398
Thr	51.6593	38.1974	9.1179
Ala	54.2126	43.7345	11.6712
Tyr	49.4836	37.9393	6.9422
His	54.2626	37.8754	11.7212
Gln	57.3956	35.7737	14.8542
Asn	59.5946	38.9238	17.0532
Lys	62.3593	36.0650	19.8179
Asp	63.6852	45.0766	21.1438
Glu	61.6780	40.0885	19.1366
Cys	48.8239	42.5501	6.2825
Trp	51.2509	40.8622	8.7095
Arg	60.9651	36.0346	18.4237
Gly	64.3824	44.8561	21.8410

Table: 4.1.1

(b): Simulation results for selection between four-column code and a new deviant code that arose due to reassignment of codon block (UUN + CUN) from Valine (Val) to Leucine (Leu).

s	Population size	Number of old codes (N_{old})	Number of deviant codes (N_d)	Cost of old code (c_{old})	Intermediate state Cost of deviant code (c_d)	Fitness of Old code (W_{old})	Fitness of deviant code (W_d)	Trials (t)	Fixation probability p_{fix}
1.0	1000	999	1	42.5414	41.1029	0.4914	0.5086	1000	0.0590
0.1	1000	999	1	42.5414	41.1029	0.9491	0.9509	1000	0.0060

Table: 4.1.2

(c) Simulation results for selection between the four column code and a new deviant code that arose due to reassignment of codons UUN and CUN from valine (Val) to isoleucine (Ile).

s	Population size	Number of old codes (N_{old})	Number of deviant codes (N_d)	Cost of old code (c_{old})	Intermediate state Cost of deviant code (c_d)	Fitness of old code (W_{old})	Fitness of deviant code (W_d)	Trials (t)	Fixation probability p_{fix}
1.0	1000	999	1	42.5414	41.6132	0.4945	0.5055	1000	0.0420
0.1	1000	999	1	42.5414	41.6132	0.9494	0.9506	1000	0.0010

Table: 4.1.3

(d): Simulation results for selection between the four column code and a new deviant code resulting from the reassignment of codons (UUN+CUN) from valine (Val) to phenylalanine (Phe).

s	Population size N	Number of old codes (N_{old})	Number of deviant codes (N_d)	Cost of old code (c_{old})	Intermediate state Cost of deviant code (c_d)	Fitness of old code (W_{old})	Fitness of deviant code (W_d)	Trials (t)	Fixation probability p_{fix}
1.0	1000	999	1	42.5414	43.7284	0.5069	0.4931	10^5	0.0000
0.1	1000	999	1	42.5414	43.7284	0.9507	0.9493	10^5	$1*10^{-4}$

Table: 4.1.4

(e): Simulation results for selection between four-column code and a new deviant code resulting from the reassignment of codons (UUN+CUN) from valine (Val) to glutamine (Gln).

s	Population size N	Number of old codes (N_{old})	Number of deviant codes (N_d)	Cost of old code (c_{old})	Intermediate state Cost of deviant code (c_d)	Fitness of old code (W_{old})	Fitness of deviant code (W_d)	Trials (t)	Fixation probability p_{fix}
1.0	1000	999	1	42.5414	57.3956	0.5743	0.4257	10^5	0.0000
0.1	1000	999	1	42.5414	57.3956	0.9574	0.9426	10^5	0.0000

Table: 4.1.5

(f): Simulation results for selection between the three codes (one is four column code, and others two are deviant codes resulting from the reassignment of codon block UUN+CUN to Ile and Leu respectively).

For s=1 and t= 10^4

Code	**Population size**	**Cost (after adoption)**	**Intermediate state Cost**	**Fitness (Intermediate state)**	**Fixation probability (p_{fix})**
Four column code	998	42.5414	------------	0.6604	$9.483*10^{-1}$
(UUN+CUN) reassigned to Ile	1	39.3419	41.6132	0.6678	$2.12*10^{-2}$
(UUN+CUN) reassigned to Leu	1	38.4918	41.1029	0.6719	$3.05*10^{-2}$

Table: 4.1.6

For a finite population size (N=10000), we took 9000 individuals with 4-column code, 500 individuals of each of the two codes (one UUN+CUN reassigned to Leu and another UUN+CUN to Ile). The variation of population structure from one generation to another is shown in the following plots.

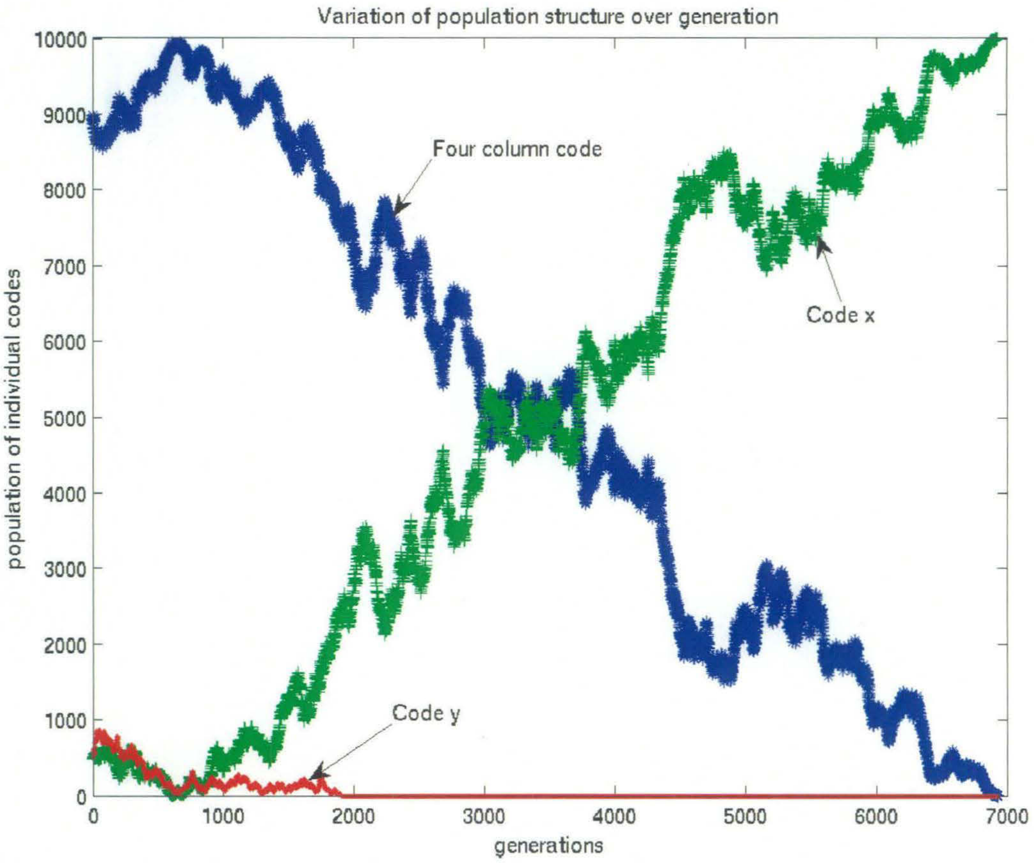

Variation of population structure over generation
population of individual codes
generations
0
1000
2000
3000
4000
5000
6000
7000
8000
9000
10000
0
1000
2000
3000
4000
5000
6000
7000
Four column code
Code x
Code y

Figure 4.1

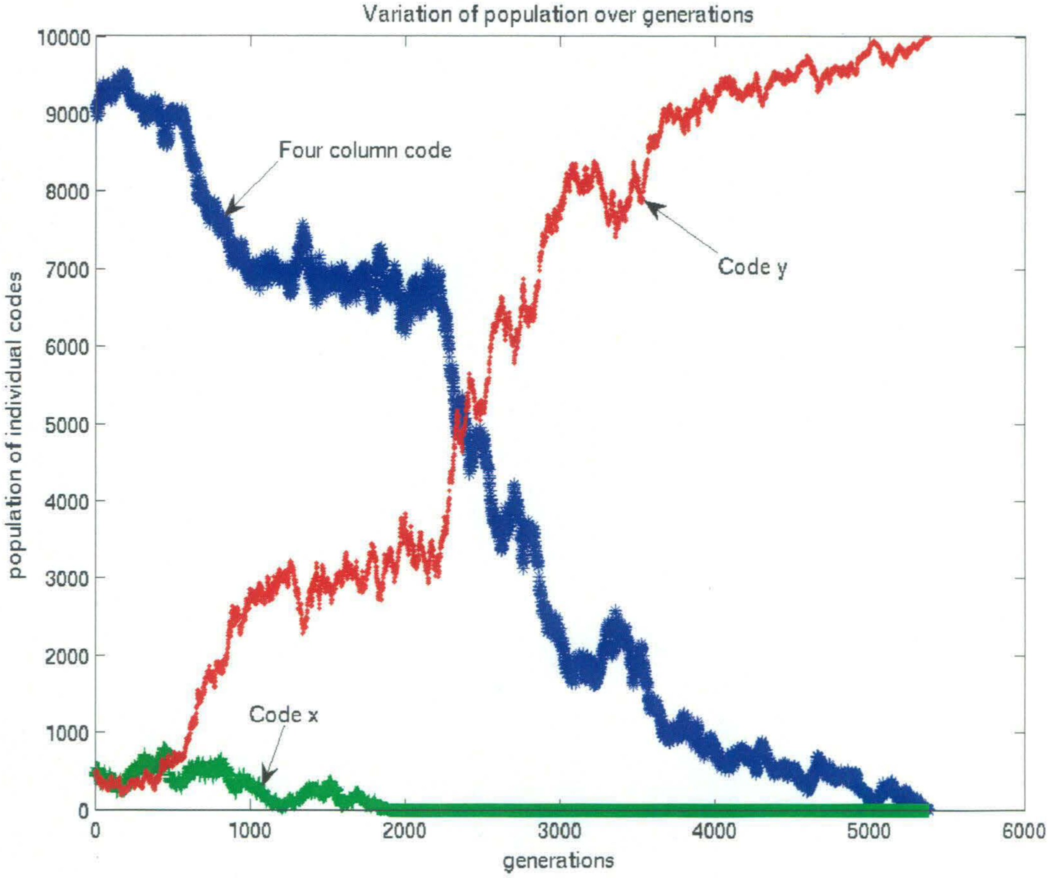


Figure 4.2

The horizontal axis represents number of generations and vertical axis represents number of individuals using the three codes: code x(green), code y(red) and four column code(blue) . Code x and code y correspond to the codes resulting from reassignment of the codon block (UUN+CUN) to the amino acids Leu and Ile respectively.

Code (with UUN+CUN reassigned to Leu) and code (with UUN+CUN reassigned to Ile) are shown to get fixed in Figure 4.1 and Figure 4.2 respectively as a result of stochasticity arising due to finite population size.

It can be seen from the Table: 4.1.1 that value of δΦ is negative i.e. intermediate state cost decreases only for the addition of amino acids leucine (Leu) and isoleucine (Ile) to the codon block (UUN+CUN). The simulation results given in Table: 4.1.4 and 4.1.5 shows that the fixation probability of a new code resulting from the addition of any amino acids except Leu and Ile to the first column starting from a 4-column code is zero. Therefore the addition any amino acids to that codon block except Leu and Ile can't be favored by the natural selection. The fixation probability of the code with codon block (UUN+CUN) reassigned to Leu (Table: 4.1.2) has the highest value. Hence we can say that Leu is the best favored amino acid to be added to that codon block by the natural selection.

The fixation probability of the deviant code with codon block (UUN+CUN) reassigned to Ile (Table 4.1.3) relative to four column code is not too much lower than the fixation probability (Table 4.1.2) of the deviant code resulting from the addition of amino acid Leu to the same codon block. Hence Ile might be added in the four column code corresponding to this codon block instead of Leu and might be incorporated in the four column code before the amino acid Leu. It is worth noting that the addition of Ile prior to Leu is also favored by the order of appearance of amino acids as Ile appears in the list of early amino acids with slightly earlier than Leu (Higgs, 2010).

As fitness of both the deviants' codes is higher (i.e. cost values are lower) than the four column code, it is possible that both the codes arose in the population of four column codes and were present in the population simultaneously. Therefore selection will occur between all these three codes. Simulation results for selection between these 3 codes are shown in Table (4.1.6) where it is clear that the fixation probability of the deviant code (UUN+CUN reassigned to Ile) is not too much lower than the fixation probability of the deviant code (UUN+CUN reassigned to Leu).

Hence it can be asserted that the code (UUN+CUN reassigned to Leu) may be outcompeted from the population (as plot 2 shows) and Ile might be incorporated in the code before Leu in the case of a finite population size which is not possible in the case of an infinite population size because the fitness of the code (UUN+CUN reassigned to Ile) is lower than the code (UUN+CUN reassigned to Leu).

(4.2): Second Column Changes

(a): The calculated cost values corresponding to different codes arised due to addition of amino acids corresponding to the codon block (ACN) in the four column code are listed in the following Table.

Cost of the four column code (Φ^{old}) = 42.5414

Amino Acids	Intermediate state cost(Φ^{int})	Cost after adoption (Φ^{new})	$\delta\Phi = (\Phi^{int} - \Phi^{old})$
Phe	47.0813	39.3289	4.5399
Leu	46.2380	40.5816	3.6966
Ile	46.7225	41.3675	4.1811
Met	45.7686	39.3541	3.2272
Val	46.0323	43.3088	3.4909
Ser	42.5615	39.8189	0.0201
Pro	43.0119	38.5289	0.4705
Thr	42.4745	37.8605	**−0.0669**
Ala	42.5414	42.5414	0.0000
Tyr	46.4063	38.0944	3.8649
His	45.1866	37.7181	2.6452
Gln	45.0257	35.4528	2.4843
Asn	44.4985	38.1634	1.9571
Lys	47.0880	35.4362	4.5466
Asp	45.9607	43.4949	3.4193
Glu	46.2164	39.5142	3.6750
Cys	44.9951	42.2546	2.4537
Trp	48.5706	40.8760	6.0292
Arg	47.7568	35.6659	5.2154
Gly	45.0594	43.0974	2.5180

Table: 4.2.1

(b): Simulation results for selection between four-column code and a deviant code resulting from the reassignment of codons ACN from Ala to Thr. It is also assumed that the other 3 columns have the same structure as the four-column code.

s	Population size N	Number of old codes (N_{old})	Number of deviant codes (N_d)	Cost of old code (c_{old})	Intermediate state Cost of deviant code (c_d)	Fitness of old code (W_{old})	Fitness of deviant code (W_d)	Trials (t)	Fixation probability p_{fix}
1.0	1000	999	1	42.5414	42.4745	0.4996	0.5004	10^5	0.0033
0.1	1000	999	1	42.5414	42.4745	0.9500	0.9500	10^5	0.0010

Table: 4.2.2

(c): Simulation results for selection between four-column code and a deviant code resulting from the reassignment of codons ACN from alanine (Ala) to serine (Ser).

s	Population size N	Number of old codes (N_{old})	Number of deviant codes (N_d)	Cost of old code (c_{old})	Intermediate state Cost of deviant code (c_d)	Fitness of old code (W_{old})	Fitness of deviant code (W_d)	Trials (t)	Fixation probability p_{fix}
1.0	1000	999	1	42.5414	42.5615	0.5001	0.4999	10^5	$5.6*10^{-4}$
0.1	1000	999	1	42.5414	42.5615	0.9500	0.9500	10^4	$8*10^{-4}$

Table 4.2.3

(d): Simulation results for selection between four-column code and a deviant code arised due to reassignment of codons ACN to (Arg).

s	Population size N	Number of old codes (N_{old})	Number of deviant codes (N_d)	Cost Of old code (c_{old})	Intermediate state Cost of deviant code (c_d)	Fitness of old code (W_{old})	Fitness of deviant code (W_d)	Trials (t)	Fixation probability p_{fix}
1.0	1000	999	1	42.5414	47.7568	0.5289	0.4711	10^5	0.0000
0.1	1000	999	1	42.5414	47.7568	0.9529	0.9471	10^5	0.0000

Table: 4.2.4

From Table 4.2.1, it can be seen that the value of $\delta\Phi$ is negative i.e. intermediate state cost decreases only for the addition of the amino acid threonine (Thr) to the codon block ACN and in all other cases it increases. The cost of addition of serine to that codon block has the second lowest (for the intermediate state) but greater than the cost of four column code.

The fixation probability of the deviant code resulting from the addition of any other amino acids except Thr to ACN block is either zero or nearly zero. Therefore, we can say that natural selection will favor the addition of Thr to the codon block ACN, which is in consistent with the fact that the amino acid Thr occurs in 2nd column of the canonical genetic code corresponding to the codon block ACN. It can be seen from the Table 4.2.1 that generally the cost of the codes resulting from the addition of those amino acids (which are present in the second column of the canonical genetic code) are lower than the others.

(4.3): Third Column Changes

(a): The calculated cost values corresponding to different codes arising due to addition of amino acids corresponding to the codons (UAR, CAR, AAR and GAR) in the four column code, are listed in the following Table. It is once again assumed that the other three columns have the same amino acid associations as that found in the four-column code.

Cost of the four column code (Φ^{old}) = 42.5414

Amino Acids	Intermediate state cost(Φ^{int})	Cost after adoption (Φ^{new})	$\Phi = (\Phi^{int} - \Phi^{old})$
Phe	55.4609	41.9491	12.9195
Leu	56.2070	44.9269	13.6656
Ile	57.6686	45.9527	15.1272
Met	52.4035	41.5036	9.8621
Val	56.7617	47.0008	14.2203
Ser	47.9633	41.1742	5.4219
Pro	46.3696	38.9479	3.8282
Thr	46.7786	38.7162	4.2372
Ala	51.8729	44.7910	9.3315
Tyr	48.8137	39.1454	6.2723
His	43.7389	37.6259	1.1975
Gln	40.6323	34.8333	**−1.9091**
Asn	40.8781	37.3238	**−1.6633**
Lys	41.8494	34.9967	**−0.692**
Asp	42.5414	42.5414	0.0000
Glu	40.5663	37.7225	**−1.9751**
Cys	55.9658	44.7753	13.4244
Trp	54.1739	42.4607	11.6325
Arg	43.6421	35.5633	1.1007
Gly	54.6502	45.1909	12.1088

Table 4.3.1

(b): Simulation results for selection between four column code and a deviant code resulting from reassignment of codons UAR,CAR,AAR and GAR to glutamic acid (Glu) where R stands for the bases A and G i.e. purines.

s	Population size N	Number Of old codes (N_{old})	Number of deviant codes (N_d)	Cost of old code (c_{old})	Intermediate state Cost of deviant code (c_d)	Fitness of old code (W_{old})	Fitness of deviant code (W_d)	Trials (t)	Fixation probability p_{fix}
1.0	1000	999	1	42.5414	40.5663	0.4881	0.5119	10^5	0.0902
0.1	1000	999	1	42.5414	40.5663	0.9488	0.9512	10^5	0.0053

Table 4.3.2

(c): Simulation results for selection between four-column code and a deviant code resulting from reassignment of codons UAR, CAR, AAR and GAR to glutamine (Gln).

s	Population size N	Number of old codes (N_{old})	Number of deviant codes (N_d)	Cost of old code (c_{old})	Intermediate state Cost of deviant code (c_d)	Fitness of old code (W_{old})	Fitness of deviant code (W_d)	Trials (t)	Fixation probability p_{fix}
1.0	1000	999	1	42.5414	40.6323	0.4885	0.5115	10^5	0.0888
0.1	1000	999	1	42.5414	0.9511	0.9489	0.9512	10^5	0.0050

Table: 4.3.3

(d): Simulation results for selection between four-column code and a deviant code arised due to reassignment of codons UAR, CAR, AAR and GAR to (Lys).

s	Population size N	Number of old codes (N_{old})	Number of deviant codes (N_d)	Cost of old code (c_{old})	Intermediate state Cost of deviant code (c_d)	Fitness of old code (W_{old})	Fitness Of deviant code (W_d)	Trials (t)	Fixation probability p_{fix}
1.0	1000	999	1	42.5414	41.8494	0.4959	0.5041	10^5	0.0321
0.1	1000	999	1	42.5414	41.8494	0.9496	0.9504	10^5	0.0022

Table 4.3.4

It can be seen from the Table 4.3.1, that there are 4 cases (addition of amino acids Glu, Gln,Asn,Lys) to the codons (UAR, CAR, AAR and GAR) where the value of $\delta\Phi$ comes out to be negative i.e. intermediate state cost decreases and all these amino acids are present in the third column of the canonical genetic code.

The fixation probability of the code (addition of Glu, see Table 4.3.2) has the highest value compared to the other cases (Tables 4.3.3, 4.3.4).

Therefore the addition of the amino acid Glu will be preferred compared to other amino acids. As the fixation probability (Table 4.3.3) of the code (addition of Gln) is not too much lower than the code (addition of Glu) , we can think of the addition of the Gln instead of Glu but the amino acid Gln is not present in the list of early amino acids (see pg.9 of Higgs 2009). Therefore it can be asserted that first of all, codon blocks in the third column of the four column code were preferably subdivided by the addition of amino acid Glu.

(4.4): Fourth Column Changes

(a): The cost of different codes resulting from the reassignment of amino acids corresponding to the codon bolock (CGN) from glycine(Gly) in the four column code, are listed in the following Table.

Cost of the four column code (Φ^{old}) = 42.5414

Amino Acids	Intermediate state cost(Φ^{int})	Cost after adoption (Φ^{new})	$\delta\Phi = (\Phi^{int} - \Phi^{old})$
Phe	45.3182	39.8061	2.7768
Leu	45.1050	42.5926	2.5636
Ile	45.2255	43.3427	2.6841
Met	44.8337	39.9554	2.2923
Val	45.0005	43.8138	2.4591
Ser	43.3081	40.1512	**0.7667**
Pro	44.1613	38.8728	1.6199
Thr	43.9787	38.5546	1.4373
Ala	43.5126	42.7970	0.9712
Tyr	45.0799	38.3511	2.5385
His	44.3998	37.9584	1.8584
Gln	44.3992	35.7364	1.8578
Asn	43.9809	38.2837	1.4395
Lys	44.9596	35.2220	2.4182
Asp	44.1992	43.0479	1.6578
Glu	44.4964	39.0689	1.9550
Cys	44.2988	42.2540	1.7574
Trp	45.7843	40.7768	3.2429
Arg	45.1621	35.6260	2.6207
Gly	42.5414	42.5414	0.0000

Table 4.4.1

(b): Simulation results for selection between four column code and a deviant code resulting from the reassignment of codon block CGN from glycine (Gly) to serine (Ser).

s	Population size N	Number of old codes (N_{old})	Number of deviant codes (N_d)	Cost of old code (c_{old})	Intermediate state Cost of deviant code (c_d)	Fitness Of old code (W_{old})	Fitness Of deviant code (W_d)	Trials (t)	Fixation probability p_{fix}
1.0	1000	999	1	42.5414	43.3081	0.5045	0.4955	10^5	0
0.1	1000	999	1	42.5414	43.3081	0.9504	0.9496	10^5	$3.4*10^{-4}$

Table 4.4.2

(c): Simulation results for selection between four-column code and a deviant code resulting from the reassignment of codons CGN from glycine (Gly) to arginine (Arg).

s	Population size N	Number of old codes (N_{old})	Number of deviant codes (N_d)	Cost of old code (c_{old})	Intermediate state Cost of deviant code (c_d)	Fitness of old code (W_{old})	Fitness of deviant code (W_d)	Trials (t)	Fixation probability p_{fix}
1.0	1000	999	1	42.5414	45.1621	0.5419	0.4851	10^5	0.0000
0.1	1000	999	1	42.5414	45.1621	0.9515	0.9485	10^5	0.0000

Table 4.4.3

(d): Simulation results for selection between four-column code and a deviant code resulting from the reassignment of codons CGN from glycine (Gly) to cysteine (Cys).

s	Population size N	Number of old codes (N_{old})	Number of deviant codes (N_d)	Cost of old code (c_{old})	Intermediate state Cost of deviant code (c_d)	Fitness of old code (W_{old})	Fitness of deviant code (W_d)	Trials (t)	Fixation probability p_{fix}
1.0	1000	999	1	42.5414	44.2988	0.5101	0.4899	10^5	0.0000
0.1	1000	999	1	42.5414	44.2988	0.9510	0.9490	10^5	$5*10^{-5}$

Table 4.4.4

It is clear from the Table 4.4.1 that addition of any amino acid to the codon block CGN leads to increase in the intermediate state cost. Out of all the 20 amino acids when serine is added to the codon block CGN, the resulting deviant code has the minimum cost. It can be seen from the standard genetic code that the amino acid serine is also encoded by the same codon block.

It is clear from the values of fixation probabilities of the new codes resulting from the addition of different amino acids (Table 4.4.2, 4.4.3, 4.4.4) that addition of serine is most favored than any other amino acids because in all other cases , fixation probability zero or very close to zero. Nevertheless, even the addition of Serine results in a substantial increase in code cost relative to the 4-column code, characterized by positive $\delta\Phi$. Hence, it seems reasonable to argue that the sub-divisions of the fourth column during code evolution occurred much later than sub-divisions of the first 3 columns and the starting point for the sub-division of the 4'th column into smaller codon blocks may not have been the 4-column code.

(4.5): Order of Addition of Early Amino Acids and Intermediate State Code Structures:

Let us suppose that the following 4 types of the deviant code arose as a result of reassignment of certain codon blocks in the four column code:

(1): Code A is the code where the codon block (UUN+CUN) is reassigned to Leu.

(2): Code B is the code where the codon block ACN is reassigned to Thr.

(3): Code C is the code where the codons (UAR, CAR, AAR and GAR) are reassigned to Glu.

(4): Code D is the code where the codon block CGN block is reassigned to Ser.

The probability of fixation of each of these 4 deviant codes can be estimated from population genetic simulations allowing for selection between these codes depending on their fitness.

The simulation results for the selection between these 5 types of the codes.

s=1, N=1000, trials(t)=10000;

Code types	**Number of codes**	**Intermediae state cost (C_{int})**	**Cost after adoption (C_{adpt})**	**Fitness in intermediate state(W_{int})**	**Number of fixations**	**Fixation probabiliy p_{fix}**
Four column code	996	42.5414	---------	0.7974	9598	0.9590
Code A	1	41.1029	38.5443	0.8043	144	$1.44*10^{-2}$
Code B	1	42.4745	37.8605	0.7977	16	$1.6*10^{-3}$
Code C	**1**	**40.5663**	**37.7225**	**0.8068**	**242**	**$2.42*10^{-2}$**
Code 4	1	43.3081	40.1512	0.7938	0	0

Table 4.5.1 a

S=0.1,N=1000,t=10000;

Code types	**Number of codes**	**Intermediae state cost (C_{int})**	**Cost after adoption (C_{adpt})**	**Fitness in intermediate state(W_{int})**	**Number of fixations**	**Fixation probabiliy p_{fix}**
Four column code	996	42.5414	---------	0.9797	9933	$9.933*10^{-1}$
Code A	1	41.1029	38.5443	0.9804	29	$2.9*10^{-3}$
Code B	1	42.4745	37.8605	0.9798	5	$5.0*10^{-4}$
Code C	**1**	**40.5663**	**37.7225**	**0.9807**	**25**	**$2.5*10^{-3}$**
Code D	1	43.3081	40.1512	0.9794	8	$8*10^{-4}$

Table 4.5.1 b

The fixation probabilities of the five different types of the codes are given in the last column of the Table (4.5.1 a, 4.5.1 b) for two different values of the parameter s. It is clear that out of the four deviant codes, code C gets fixed with highest probability (because its cost is lowest) i.e. it outcompetes other types (deviant codes) with the largest probability.

Thus we observe that the code C (third column changes) is the most favored compared to the other deviant codes and it is therefore most likely to replace the four column code, if all the four deviant codes arise in the population. It is also favored by the order of appearance of amino acids as Glu is one of the earliest amino acids (Higgs, 2010).

Let us suppose that after fixation of the code C (in Table 4.5.1a and 4.5.1b) which resulted from the addition of amino acid Glu through its association with the codons (UAR, CAR, AAR and GAR) in the four column code, the following new codes appeared in the population of code 3.

(1): Code 1 is the code that results from the association of Leu to the codon block (UUN+CUN) in the code C.

(2): Code 2 is the code that arises as a result of addition of Ser to the codon block UCN in the code C.

(3): Code 3 is the code that arises as a result of addition of Pro to the codon block CCN in the code C.

(4): Code 4 is the code that arises as a result of addition of Thr to the codon block ACN in the code C.

(5): Code 5 is the code that arises as a result of addition of Ile to the codon block AUN in the code C.

Selection between all these codes newly originated codes will eventually lead to the fixation of one of the deviant codes indicating the next stage of the code evolution process.

The simulation results corresponding to this have been tabulated below

s=1, N=1000, t=10000;

Code types	**Number of codes**	**Intermediae state cost (C_{int})**	**Cost after adoption (C_{adpt})**	**Fitness in intermediate state(W_{int})**	**Number of fixations**	**Fixation probabiliy p_{fix}**
Code C	995	37.7225	----------	0.8322	9770	$9.77*10^{-1}$
Code 1	**1**	**36.2841**	**33.7255**	**0.8386**	**151**	$\mathbf{1.51*10^{-2}}$
Code 2	1	37.7426	35.0376	0.8329	9	$9*10^{-4}$
Code 3	1	38.1931	33.9214	0.8301	0	0
Code 4	1	37.6557	33.3294	0.8325	22	$2.2*10^{-3}$
Code 5	1	37.2584	34.6562	0.8343	48	$4.8*10^{-3}$

Table 4.5.2 a

S=0.1,N=1000,t=10000;

Code types	**Number of codes**	**Intermediae state cost (C_{int})**	**Cost after adoption (C_{adpt})**	**Fitness in intermediate state(W_{int})**	**Number of fixations**	**Fixation probabiliy p_{fix}**
Code C	995.0000	37.7225	----------	0.9832	9930.0000	$9.93*10^{-1}$
Code 1	**1.0000**	**36.2841**	**33.7255**	**0.9839**	**20.0000**	$\mathbf{2.0*10^{-3}}$
Code 2	1.0000	37.7426	35.0376	0.9832	10.0000	$1.0*10^{-3}$
Code 3	1.0000	38.1931	33.9214	0.9830	12.0000	$1.2*10^{-3}$
Code 4	1.0000	37.6557	33.3294	0.9833	13.0000	$1.3*10^{-3}$
Code 5	1.0000	37.2584	34.6562	0.9834	15.0000	$1.5*10^{-3}$

Table 4.5.2 b

The fixation probabilities of six different types of the codes are given in the last column of the above Table for the two values of the selection parameter s. It can be seen that out of the five deviant codes, code 1 has the highest fixation probability. Hence if all the five types of the deviant codes are present in the population primarily consisting of code 3, then the new deviant code 1 will be most likely to be fixed in the population at the next stage of the code evolution process..

Suppose that after fixation of the code 1, the following new codes appear by chance in the population of code 1.

(1): Code 6 results from the addition of Ile to the codon block AUN in the code 1.
(2): Code 7 results from the addition of Ser to the codon block UCN in the code 1.
(3): Code 8 results from the addition of Pro to the codon block CCN in the code 1.
(4): Code 9 results from the addition of Thr to the codon block ACN in the code 1.

The simulation results below indicate which of the above deviant codes is most likely to be fixed in the next stage of code evolution process, for two different values of the selection coefficient.

s=1, N=1000, t=10000;

Code types	**Number of codes**	**Intermediae state cost (C_{int})**	**Cost after adoption (C_{adpt})**	**Fitness in intermediate state(W_{int})**	**Number of fixations**	**Fixation probabiliy p_{fix}**
Code 1	996	33.7255	--------	0.8013	9982	$9.98*10^{-1}$
Code 6	1	34.3970	33.0540	0.7974	0	0
Code 7	1	33.7647	31.0609	0.8011	8	$8*10^{-4}$
Code 8	1	34.1812	30.1538	0.7986	0	0
Code 9	**1**	**33.6881**	**29.7796**	**0.8016**	**10**	**$1*10^{-3}$**

Table 4.5.3 a

S=0.1,N=1000,t=10000;

Code types	**Number of codes**	**Intermediae state cost (C_{int})**	**Cost after adoption (C_{adpt})**	**Fitness in intermediate state(W_{int})**	**Number of fixations**	**Fixation probabiliy p_{fix}**
Code 1	996	33.7255	--------	0.9801	9959	$9.959*10^{-1}$
Code 6	1	34.3970	33.0540	0.9797	8	$8*10^{-4}$
Code 7	1	33.7647	31.0609	0.9801	12	$1.2*10^{-3}$
Code 8	1	34.1812	30.1538	0.9799	6	$6*10^{-4}$
Code 9	**1**	**33.6881**	**29.7796**	**0.9802**	**15**	**$1.5*10^{-3}$**

Table 4.5.3 b

Therefore we can assert that the code structure the follows the fixation of code 1 is most likely to be the code 9.

Let us now consider that after fixation of the code 9 (Tables 4.5.3a and 4.5.3b) which arose from addition of Thr to the codon block ACN in the code 1 (section 4.5.2), following new codes are originated in the population of the code 9.

(1): Code 10 is the code originated when Ser was added to the codon block UCN in the code 9.
(2): Code 11 is the code originated when Pro was added to the codon block CCN in the code 9.
(3): Code 12 is the code originated when Ile was added to the codon block AUN in the code 9.

The population structure will evolve as a result of competition between the codes. The simulation results for this are as follows:

s=1, N=1000, t=10000;

Code types	Number of codes	Intermediae state cost (C_{int})	Cost after adoption (C_{adpt})	Fitness in intermediate state(W_{int})	Number of fixations	Fixation probabiliy p_{fix}
Code 9	997	29.7796	---------	0.7556	9999	$9.999*10^{-1}$
Code 10	**1**	**29.9855**	**27.6546**	**0.7539**	**1**	$\mathbf{1.0*10^{-4}}$
Code 11	1	31.6020	29.0456	0.7406	0	0
Code 12	1	30.4597	29.1044	0.7500	0	0

Table 4.5.4 a

S=0.1,N=1000,t=10000;

Code types	Number of codes	Intermediae state cost (C_{int})	Cost after adoption (C_{adpt})	Fitness in intermediate state(W_{int})	Number of fixations	Fixation probabiliy p_{fix}
Code 9	997.0000	29.7796	---------	0.9756	9980.0000	$9.99*10^{-1}$
Code 10	**1.0000**	**29.9855**	**27.6546**	**0.9754**	**10.0000**	$\mathbf{1.0*10^{-3}}$
Code 11	1.0000	31.6020	29.0456	0.9741	2.0000	$2.0*10^{-4}$
Code 12	1.0000	30.4597	29.1044	0.9750	8.0000	$8.0*10^{-4}$

Table 4.5.4 b

Therefore it is reasonable to infer that the code structure that follows the fixation of code 9 is most likely to be the code 10.

Consider that after fixation of the code 10 which arose from addition of the amino acid Ser to the codon block UCN in the code 9, following new codes appeared in the population of code 10.

(1): Code 13 is the code resulting from the reassignment of the codons (CCN) in the code 10 to the amino acid Pro.

(2): Code 14 is the code resulting from the reassignment of the codons (AUN) in the code 10 to the amino acid Ile.

As there are two deviant codes present in the population of the codes 10, selection will occur among all these three codes including the original codes (code 10).

s=1, N=1000, t=10000;

Code types	**Number of codes**	**Intermediae state cost (C_{int})**	**Cost after adoption (C_{adpt})**	**Fitness in intermediate state(W_{int})**	**Number of fixations**	**Fixation probabiliy p_{fix}**
Code 10	998	27.6546	---------	0.6780	10000	1
Code 13	1	29.9031	26.8427	0.6519	0	0
Code 14	**1**	**28.3347**	**26.9559**	**0.6701**	**0**	**0**

Table 4.5.5 a

S=0.1,N=1000,t=10000;

Code types	**Number of codes**	**Intermediae state cost (C_{int})**	**Cost after adoption (C_{adpt})**	**Fitness in intermediate state(W_{int})**	**Number of fixations**	**Fixation probabiliy p_{fix}**
Code 10	998	27.6546	---------	0.9678	9996	$9.996*10^{-1}$
Code 13	1	29.9031	26.8427	0.9652	0	0
Code 14	**1**	**28.3347**	**26.9559**	**0.9670**	**4**	**$4.0*10^{-4}$**

Table 4.5.5 b

The fixation probabilities of the two deviant codes are given in the last column of the above Table for the two values of the parameter s. From the values of the fixation probabilities, it is clear that the fixation probability of the deviant code 13 is equal to 0 in both the cases(s=1 and s=0.1) but the deviant code 14 is being fixed with the some lower probability. If these two deviant codes arise and present in the population of the original codes (code 10) at a time, most probably the deviant code 14 will invade the whole population and code 13 will be eliminated from the population.

Therefore it can be observed that the code structure after the fixation of code 10 is most likely to be the code 14 and it can also be concluded that the amino acid Pro might have been incorporated in the code corresponding to the codon block UCN after the fixation of the code 14.

In this way the early amino acids (Glu, Leu, Thr, Ser, Ile, Pro) which were not present in the four column code can be added following the most probable pathway shown below by subsequent subdivision of codon blocks to get the code structure where all the early amino acids (see page 16, Higgs 2009) have been incorporated in the four column code.

	U	C	A	G	
U	Val	Ala	Asp	Gly	U C
					A G
C					U C
					A G
A					U C
					A G
G					U C
					A G

(1): Four column code

	U	C	A	G	
U	Val	Ala	Asp	Gly	U C
			Glu		A G
C			Asp		U C
			Glu		A G
A			Asp		U C
			Glu		A G
G			Asp		U C
			Glu		A G

(2): Code C

	U	C	A	G	
U	**Leu**	Ala	Asp	Gly	U C
			Glu		A G
C			Asp		U C
			Glu		A G
A	Val		Asp		U C
			Glu		A G
G			Asp		U C
			Glu		A G

(3): Code 1

	U	C	A	G	
U	Leu	Ala	Asp	Gly	U C
			Glu		A G
C			Asp		U C
			Glu		A G
A	Val	**Thr**	Asp		U C
			Glu		A G
G		Ala	Asp		U C
			Glu		A G

(4): Code 9

	U	C	A	G	
U	Leu	**Ser**	Asp	Gly	U C
			Glu		A G
C		Ala	Asp		U C
			Glu		A G
A	Val	Thr	Asp		U C
			Glu		A G
G		Ala	Asp		U C
			Glu		A G

(5): Code 10

	U	C	A	G	
U	Leu	Ser	Asp	Gly	U C
			Glu		A G
C		Ala	Asp		U C
			Glu		A G
A	**Ile**	Thr	Asp		U C
			Glu		A G
G	Val	Ala	Asp		U C
			Glu		A G

(6): Code 14

	U	C	A	G	
U	Leu	Ser	Asp	Gly	U C
			Glu		A G
C		Pro	Asp		U C
			Glu		A G
A	Ile	Thr	Asp		U C
			Glu		A G
G	Val	Ala	Asp		U C
			Glu		A G

(7) The code structure after incorporation of the 10 earliest amino acids

(4.6): An alternative pathway based on the order of addition of early amino acids and intermediate state code structures:

It can be seen from the Tables (4.5.3 a and 4.5.3 b) that the fixation probability of the code 9 is only slightly greater (more generally comparable to the code 7) than the fixation probability of the code 7. Hence it is reasonable to expect that the code 7 might have invaded the whole population instead of code 9. Therefore the code 7 can be supposed to be the next code structure to be fixed after the fixation of the code 1.

Let us consider that after fixation of the code 7(Tables 4.5.3 a and 4.5.3 b) which resulted from addition of Ser to the codon block UCN in the code 1 , following new codes appeared in the population of the code 7.

(1): Code 15 resulted from addition of Pro to the codon block CCN in the code 7.
(2): Code 16 resulted from addition of Thr to the codon block ACN in the code 7.
(3): Code 17 arose as a result of addition of Ile to the codon block AUN in the code 7.

Selection will result among all these codes including the original codes (code 7), if all these three deviant codes are present at a time in the population of the original codes.

The simulation results for the selection between the above codes including the code 7 are given below.

s=1, N=1000, t=100000;

Code types	Number of codes	Intermediate state cost (C_{int})	Cost after adoption (C_{adpt})	Fitness in intermediate state(W_{int})	Number of fixations	Fixation probability p_{fix}
Code 7	997	31.0609	---------	0.7563	100000	1
Code 15	1	32.4852	27.9276	0.7451	0	0
Code 16	1	32.1917	27.6546	0.7474	0	0
Code 17	**1**	**31.7234**	**30.3569**	**0.7511**	**0**	**0**

Table 4.6.1 a

S=0.1,N=1000,t=10000;

Code types	Number of codes	Intermediate state cost (C_{int})	Cost after adoption (C_{adpt})	Fitness in intermediate state(W_{int})	Number of fixations	Fixation probability p_{fix}
Code 7	997	31.0609	---------	0.9756	99901	$9.9901*10^{-1}$
Code 15	1	32.4852	27.9276	0.9745	16	$1.6*10^{-4}$
Code 16	1	32.1917	27.6546	0.9747	33	$3.3*10^{-4}$
Code 17	**1**	**31.7234**	**30.3569**	**0.9751**	**50**	**$5.0*10^{-4}$**

Table 4.6.1 b

The intermediate state cost values (third column of the above Table) of these three deviant codes are higher than the cost of the original code. The fixation probabilities of all these deviant codes comes out to be 0 for s=1 i.e. no one of the deviant codes is getting fixed even for a single time in 10^5 trials. From the Table 4.6.1 b, it is evident that out of these three deviant codes, code 17 has the highest fixation probability for s=0.1. Hence if all these three deviant codes originated at a time in the population which primarily consists of the code 9 then deviant code 17 will be the most dominant compared to other deviant codes to invade the whole population.

Therefore we can assert that the code structure after the fixation of the code 7 is most likely to be the code 17.

Let us consider that after fixation of the code 17 which arose from addition of the amino acid Ile to the codon block AUN in the code 7, following new codes resulted and present at a time in the population of the original codes (code 17).

(1): Code 18 arose as a result of reassignment of the codons (CCN) in the code 17 to the amino acid Pro.

(2): Code 19 is resulted from the reassignment of the codons (ACN) in the code 17 to the amino acid Thr.

Selection between all these codes including the code 17 finally leads to the fixation of one of them.

Simulation results for this are given in the following Table.

s=1, N=1000, t=100000;

Code types	**Number of codes**	**Intermediate state cost (C_{int})**	**Cost after adoption (C_{adpt})**	**Fitness in intermediate state(W_{int})**	**Number of fixations**	**Fixation probability p_{fix}**
Code 17	998	30.3569	--------	0.6757	10^5	1
Code 18	1	31.7769	27.2238	0.6605	0	0
Code 19	**1**	**31.4754**	**26.9559**	**0.6638**	**0**	**0**

Table 4.6.2 a

S=0.1,N=1000,t=100000;

Code types	**Number of codes**	**Intermediate state cost (C_{int})**	**Cost after adoption (C_{adpt})**	**Fitness in intermediate state(W_{int})**	**Number of fixations**	**Fixation probability p_{fix}**
Code 17	998	30.3569	--------	0.9676	99964	$9.9964*10^{-1}$
Code 18	1	31.7769	27.2238	0.9966	14	$1.4*10^{-4}$
Code 19	**1**	**31.4754**	**26.9559**	**0.9664**	**22**	**$2.2*10^{-4}$**

Table 4.6.2 b

From the intermediate state cost values and their fixation probabilities of the two deviants' codes, we can conclude that the code 19 is the more efficient than the code 18 to be fixed in the population. From the values of the fixation probabilities given in the above Table, it can be asserted that the most probable code structure after the fixation of the code 17 is the code 19 and amino acid Pro might have been added to the code corresponding to the codon block UCN after the fixation of the code 19.

In this case the early amino acids (Glu, Leu, Thr, Ser, Ile, Pro) which were not present in the four column code might be incorporated by the subsequent division of codon blocks to get the intermediate code structure where all the early amino acids (Higgs, 2009) have been incorporated in the four column code following the pathway (which is second most probable pathway) shown below.

	U	C	A	G	
U	Val	Ala	Asp	Gly	U C
					A G
C					U C
					A G
A					U C
					A G
G					U C
					A G

(1): four column code

	U	C	A	G	
U	Val	Ala	Asp	Gly	U C
			Glu		A G
C			Asp		U C
			Glu		A G
A			Asp		U C
			Glu		A G
G			Asp		U C
			Glu		A G

(2): Code C

	U	C	A	G	
U	**Leu**	Ala	Asp	Gly	U C
			Glu		A G
C			Asp		U C
			Glu		A G
A	Val		Asp		U C
			Glu		A G
G			Asp		U C
			Glu		A G

(3): Code 1

	U	C	A	G	
U	Leu	**Ser**	Asp	Gly	U C
			Glu		A G
C		Ala	Asp		U C
			Glu		A G
A	Val		Asp		U C
			Glu		A G
G			Asp		U C
			Glu		A G

(4): Code 7

	U	C	A	G	
U	Leu	Ser	Asp	Gly	U C
			Glu		A G
C		Ala	Asp		U C
			Glu		A G
A	**Ile**		Asp		U C
			Glu		A G
G	Val		Asp		U C
			Glu		A G

(5): Code 17

	U	C	A	G	
U	Leu	Ser	Asp	Gly	U C
			Glu		A G
C		Ala	Asp		U C
			Glu		A G
A	Ile	**Thr**	Asp		U C
			Glu		A G
G	Val	Ala	Asp		U C
			Glu		A G

(6): Code 19

	U	C	A	G	
U	Leu	Ser	Asp	Gly	U C
			Glu		A G
C		**Pro**	Asp		U C
			Glu		A G
A	Ile	Thr	Asp		U C
			Glu		A G
G	Val	Ala	Asp		U C
			Glu		A G

(7): **Code structure after incorporation of 10 early amino acids**

The existence of the two alternative pathways of code evolution are possible because of the possibility of fixation of code 7, which has a slightly higher cost than code 9, in a finite population model. The fixation of code 7 instead of code 9 changes the subsequent evolutionary trajectories of the genetic code even though the final code incorporating the 10 earliest amino acids is still the same as before.

(4.7): Consistency of the code evolution pathway determined by (i) Co-evolution theory and (ii) Physico-chemical theory

The co-evolution theory is an alternative proposal for determining the pathways of code evolution in the early stages prior to the advent of the Last Universal Common Ancestor (LUCA) of all living organisms. Di-Giluio *et al.* (1999) proposed a specific pathway for code evolution based on the gradual ceding of codon blocks prom precursor (abiologically synthesized) amino acids to product amino acids i.e. those that require precursor amino acids for their bio-synthesis. In this section, we have estimated the intermediate cost of the various stages of code as postulated by Di-giulio, starting from the previous stage. For example, Code (2) below is obtained from reassignment of certain codon blocks of Code (1). As mentioned earlier, the intermediate state cost corresponds to the cost of the code immediately after reassignment and before the codon have had a chance to adapt to the new code. The cost after adoption is the cost of the code after the codons have equilibrated following reassignment of codon blocks.

Second base

First base

Ser Ala	Ser	Ser Ter	Ser Ter Ser
	Glu	Glu	Glu
Asp	Asp	Asp	Gly Glu
Ala	Ala	Asp Glu	Gly

(1)

Cost = 20.7394

Second base

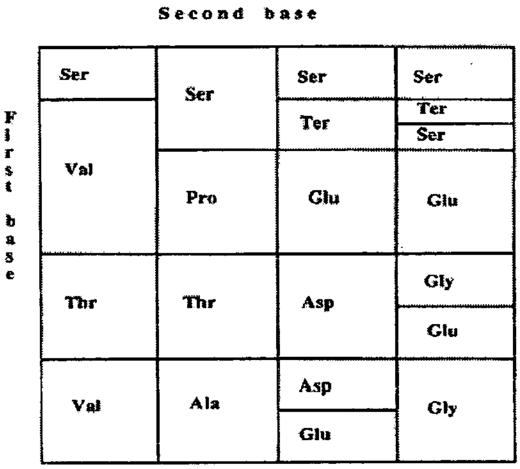
First base

Ser Val	Ser	Ser Ter	Ser Ter Ser
	Glu	Glu	Glu
Asp	Asp	Asp	Gly Glu
Val	Ala	Asp Glu	Gly

(2)

cost (intermediate state) = 25.4967

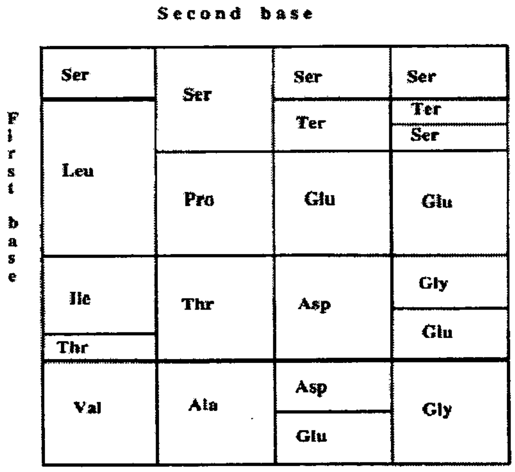
cost (after adoption) = 20.1225



(3)

cost (intermediate state) = 23.2324

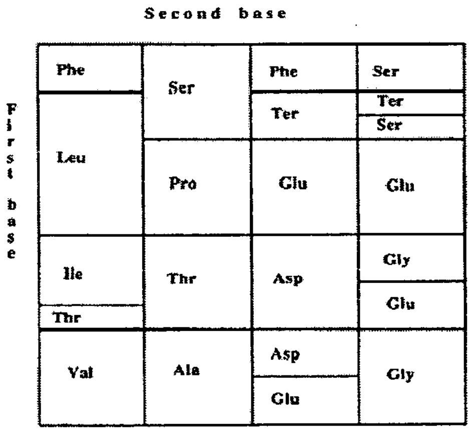
cost (after adoption) = 19.2714



(4)

cost (intermediate state) = 21.5430

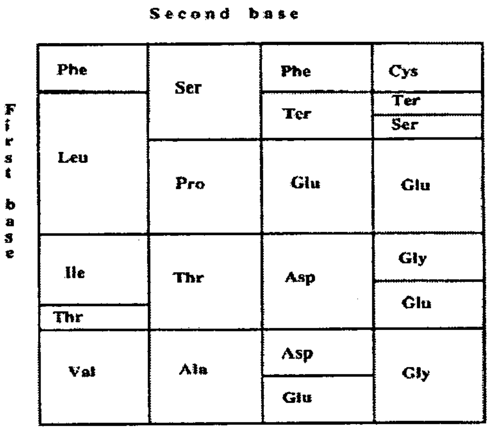
cost (after adoption) = 15.8059



(5)

cost (intermediate state) = 18.5791

cost(after adoption) = 15.6380



(6)

cost (intermediate state) = 17.5648

cost(after adoption) = 16.2686

Second base

First base

Phe	Ser	Tyr Ter	Cys Ter Ser
Leu	Pro	Glu	Glu
Ile Thr	Thr	Asp	Ser Glu
Val	Ala	Asp Glu	Gly

(7)

cost (intermediate state) = 16.2247

cost(after adoption) = 15.1142

Second base

First base

Phe	Ser	Tyr Ter	Cys Ter Ser
Leu	Pro	Glu	Arg
Ile Thr	Thr	Asp Lys	Ser Arg
Val	Ala	Asp Glu	Gly

(8)

cost (intermediate state) = 15.4573

cost(after adoption) = 15.4573

Second base

First base

Phe	Ser	Tyr Ter	Cys Ter Ser
Leu	Pro	Gln	Arg
Ile Thr	Thr	Asn Lys	Ser Arg
Val	Ala	Asp Glu	Gly

(9)

cost (intermediate state) = 15.3018

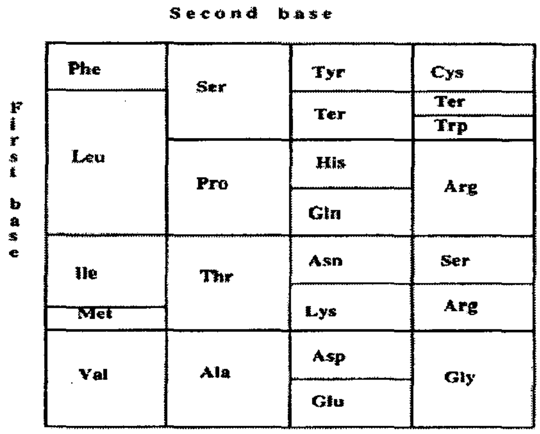
cost(after adoption) = 15.3018

Phe	Ser	Tyr Ter	Cys Ter Ser
Leu	Pro	His Gln	Arg
Ile Thr	Thr	Asn Lys	Ser Arg
Val	Ala	Asp Glu	Gly

(10)

cost (intermediate state) = 15.2791

cost(after adoption) = 15.2791



(11)

cost (intermediate state) = 17.1823

cost(after adoption) = 15.278

The fixation probability of a primordial code (encoding less than 20 amino acids) competing with a population of (N-1) other codes belonging to the previous stage is estimated below for a few of the above cases.

(b): Simulation Results for selection between code (2) and code (3)

s	Populatin size N	Number of codes (2) (N_{old})	Number of deviant code (3) (N_d)	Fitness of old code(2) (W_{old})	Fitness of deviant code(3) (W_d)	Trials (t)	Fixation probability p_{fix}
1.0	**1000**	**999**	**1**	**0.5359**	**0.4641**	**10^5**	**0.0000**
0.1	**1000**	**999**	**1**	**0.9536**	**0.9464**	**10^5**	**$1*10^{-5}$**

Table 4.7.1

(c): Simulation Results for selection between code (3) and code (4)

s	Populatin size N	Number of codes (3) (N_{old})	Number of deviant code (4) (N_d)	Fitness of old code(3) (W_{old})	Fitness of deviant code(4) (W_d)	Trials (t)	Fixation probability p_{fix}
1.0	1000	999	1	0.5278	0.4722	10^5	0.0000
0.1	1000	999	1	0.9528	0.9472	10^5	$1*10^{-5}$

Table 4.7.2

It is clear that the intermediate state cost of each of the codes is greater than the cost (after adoption) of the code structure in its previous state except in the two cases (code 9 and 10). The intermediate state cost of the code (9) is slightly lower than the final state cost of code (8) and the intermediate state cost of the code (10) is slightly lower than the final state cost of code (9). Therefore, in general the fitness (in intermediate state) of the each code will be lower than the fitness of the code structure in its previous state.

As discussed in chapter 3, in an infinite population size a single deviant code with lower fitness can't be fixed, hence in a case of infinite population size the code with higher intermediate state cost i.e with lower fitness will be eliminated from the population. Even for finite populations, the fixation probability of deviant codes dictated by the coevolution theory is nearly zero, as shown in Tables 4.7.1, 4.7.2. From the above observations we can infer that that the pathway for the evolution of the genetic code, given according to the coevolution theory is not consistent with the physicochemical hypothesis (where evolutionary pressure to structure the code as standard code is driven by the reduction in the cost due to replacement of a amino acid by another due to code expansion and/or mistranslational errors).

Chapter 5
CONCLUSIONS AND FUTURE DIRECTIONS

Generally, a new amino acid can be added to a specific codon block (as a result of reassignment of codons from a more ancient to the newer amino acid) in the early stage of evolution of the code structure. Even though the new amino acid can be added to any codon block, the code that is likely to be fixed has the lowest cost (i.e. maximum fixation probability). It is also interesting to note that new code which is fixed after reassignment of a codon block from an older to a newer amino acid is the one in which the position of new amino acid encoded is consistent with its position in the canonical code.

We have analyzed the pathway of early evolution of the genetic code by calculating the fixation probabilities of the possible alternative codes (whose intermediate state cost was lower or not too much greater than the previous code) for a finite population size and explored mainly the following points:

(1): (a): The fixation probability (see the Tables 4.1.2, 4.1.3) of the code resulting from the reassignment of the codon block (UUN+CUN) from amino acid Val to Ile in the four column code is not much lower than the code resulting from addition of Leu to the same codon block. Therefore, we conclude that Ile might be added to that codon block and incorporated in the code before Leu which will lead to change the pathway for the early evolution of the code. This conclusion is favored by the order of appearance of amino acids as Ile appears ahead of Leu in the list of early amino acids (Higgs, 2009).

(b): Thr is found to be the most favored amino acid to be added to the codon block ACN in the second column of the four column code because the code resulting from this has the highest fixation probability.

Since the fixation probabilities of the alternative codes resulting from the addition of other amino acids to the same codon block is very near to zero, hence even in the presence of the stochasticity resulting from the finite population size, natural selection cannot favor the addition of other amino acids to that codon block.

(c): In the third column of the four column code Glu is the amino acid which when added to the codons (UAR, CAR, AAR and GAR) in the 4-column code results in a new code that has the highest fixation probability (see the Table 4.3.2).This indicates that the addition of Glu corresponding to those codons is most favored by natural selection. This is also bolstered by the order of appearance of early amino acids, as Glu is one of the top five amino acids in the list of 10 earliest amino acids (see pg.9 of Higgs, 2009).

As the intermediate state cost of the code (addition of Gln to the same codon blocks as above) is also lower than the four column code and fixation probability (see Table 4.3.3) of this code is only slightly lower than the previous code (resulting from addition of Glu), the addition of Gln could be assumed to be reasonable instead of Glu for those codon blocks. However, Gln is not present in the list of early amino acids (see pg.9 of Higgs, 2009). This leads us to conclude that the codon blocks in the third column of the four column code might be preferably subdivided by the addition of amino acid Glu instead of Gln.

(d): The addition of any amino acid corresponding to the codon block CGN in the fourth column of the four column code leads to increase in the intermediate state cost (Table 4.4.1) but it is minimum for the addition of Ser.

Hence, it is reasonable to conclude that fourth column of the four column code might have been subdivided after code resulting from the subdivisions in other columns of the four column code had become fixed in the population. Nevertheless, when the CGN block is eventually reassigned from Gly, it may have been reassigned to the amino acid Ser.

(2): We identified the order of addition of the remaining early amino acids (which were not present in the four column code) to get the code structure (see pg.15 of Higgs, 2009) in which all the early amino acids have been incorporated.

We calculate the fixation probabilities of all possible alternative codes (which are consistent with structure of the canonical code) resulting from addition of early amino acids to the codon blocks of the column 1, column 2 and column 3 in the four column code. The fixation probability of the code resulting from the addition of Glu corresponding to the codon bocks came out to be maximum. Therefore it is reasonable to conclude that first of all the third column was subdivided by the addition of Glu.

Next we started from the code where Glu was added corresponding to codons (UAR, CAR, AAR and GAR) and calculated the fixation probabilities of all the possible alternative codes resulting from the addition of early amino acids (see the Tables 4.5.1 a and 4.5.1 b) and found that the code resulting from the addition of Leu corresponding to the codon block (UUN+CUN) in the code where Glu has been already incorporated, got fixed with highest probability. This lead us to conclude that Leu was incorporated in the code after Glu.

In the same way we calculated the fixation probability of the various alternative codes (see Tables 4.5.2 a, 4.5.2 b) resulting from the addition of an amino acid in the code where Glu and Leu have already been added. The code with Thr corresponding to the codon block ACN appeared with the highest fixation probability. Therefore we conclude that most probably Thr might have been added to the codon block ACN after the addition of amino acids Glu and Ile in the four column code.

Following a similar procedure as above it we are able to predict the intermediate state code structures (given in the section 4.5) which lead us from four column code to the code where all the early amino acids have been incorporated. Thus the most probable order of addition of early amino acids (which were not present in the four column code) is as follows:

Glu ,Leu, Thr, Ser, Ile, Pro

It is remarkable to see that the fixation probability of the code resulting from the addition of Ser to the codon block UCN in the code where Glu and Leu has been already added in the four column code is very slightly lower (see the Tables 4.5.3 a and 4.5.3 b) than the code resulting from addition of Thr to the codon block UCN in the code where two early amino acids Glu and Leu has already been incorporated in the four column code. Therefore it is also reasonable to expect that this code might have been fixed and other alternative codes eliminated from the population.

In this case, by following the similar procedure as above we are able to predict the new pathway (see the section 4.6) for the early evolution of the code to get the code where all the 10 early amino acids have been incorporated. This pathway can be thought of as the second most probable pathway which might result from the effect of the stochasticity arising as a result of finite population size. In this second most probable pathway the order of addition of the early amino acids (which were not present in the four column code) in the four columns code is as follows:

Glu, Leu, Ser, Ile, Thr, Pro

(3): The values of the cost (in the intermediate state) of each of the intermediate state code structures in the pathway predicted by the coevolution theory generally comes out to be greater than the cost (after adoption) of its code structure in the previous evolutionary stage (see the section 4.7). Even for finite populations, the fixation probability of deviant codes (dictated by the coevolution theory) is nearly zero, as shown in Tables 4.7.1 and 4.7.2. Thus it is reasonable to conclude that the pathway for the evolution of the genetic code predicted by coevolution theory is not consistent with the pathway predicted by the physicochemical hypothesis.

Future Directions:

(1): In this thesis we have considered the selection among the individuals with different codes until one of them invade the whole population, ignoring the rise of new codes during this period. A new code may arise as a result of reassignment of some codons to another amino acid in one or more of the previous codes during the selection. Therefore, it may be fruitful to see how the structure of population changes from one generation to another by considering the evolution of among different codes in which both selection and mutation (codon reassignments which lead to the appearance new codes) play a role.

(2): In a population of various types of codes, each code will compete to invade the population by translating its gene sequences. As there are also mutations occurring in the gene sequences, therefore it will be more realistic to study the coevolution of the genetic code and protein-coding genes (code-message evolution) constrained by the stochasticity arising as a result of finite population size.

(3): In this thesis we calculated the fitness of the individual codes which was independent of their frequency in the population. It may be interesting to consider the case where fitness is dependent on the frequency of different codes in the population using a game theoretic evolutionary model. Such frequency dependent selection may have been relevant in the very early stages of the evolution of the genetic code.

REFERENCES

(1): Creighton TE: **Proteins: structures and molecular properties.** *W.H. Freeman, New York;* 1993.

(2): Cric FH: **The origin of genetic code.** *J. Mol. Bio.* 1968, **38(3)**:367-379.

(3): Di Giulio M and Medungo M: **Physicochemical optimization in the genetic code origin as the number of codified amino acids increase.** *J Mol Evol,* 1999, **49**:1-10.

(4): Di Giulio M: **On the origin of genetic code.** *J. Theor. Biol.* 1997, **187**:573-581.

(5): Di Giulio M: **An extension of the coevolution theory of the origin of genetic code.** *Biology Direct* 2008, **3**:37.

(6): Dillon LS: **The origins of genetic code.** *Bot. Rev.* 1973, **39**:301-345.

(7): Engleman DA, Steitz TA and Goldman A: **Identifying non-polar transbilayer helices in amino acid sequences of membrane proteins.** *Annu Rev Biophys Chem* 1986, **15**:321-353.

(8): Freeland SJ, Hurst LD: **The genetic code is one in a million.** *J Mol Evol* 1998, **47**:238-248.

(9): Friedman SM, Weinstein IB: **Lack of fidelity in the translation of ribopolynucleotides.** *Proc Natl Acad Sci USA* 1964, **52**:988-996.

(10): Gilis D, Massar S, Cerf NJ and Rooman M: **Optimality of the genetic code with respect to protein stability and amino-acid frequencies.** *Genome Biology* 2001, **2(11)**:49.1-49.12.

(11): Higgs PG, Hao W and Golding GB: **Identification of selective effects on highly expressed genes.** *Evol Bioinform Online* 2007, **3**:1-13.

(12): King JL, Jukes TH: **Non-Darwinian evolution.** *Science* 1969, **164**:788-798.

(13): Knight RD, Freeland SJ, Landweber LF: **Rewiring the keyboard: evolvability of the genetic code.** *Nat Rev Genet* 2001, **2**:49-58.

(14): Kyte J and Doolittle RF: **A simple method for displaying the hydropathic character of a protein.** *J Mol Bio* 1982, **157**:105-132.

(15): Millers S, Janin J, Lesk AM and Chothia C: **Interior and surface of monomeric proteins.** *J Mol Bio* 1987, **196**:641-657.

(16): Nirenberg MW, Jones OW, Leder P, Clark Bfc,Sly WS and Pestka S : **On the coding of genetic information.** *Cold Spring Harbor Symp. Quant. Biol.* 1963, **28**: 549-557.

(17): Parker J: **Errors and alternatives in reading the universal genetic code.** *Microbiol Rev 1989*, **53**:273-298.

(18): Paul G Higgs: **A four-column theory for the origin of the genetic code: tracing the evolutionary pathways that gave rise to an optimized code.** *Bilology Direct* 2009, **4**:16.

(19): Pelc SR : **Correlation between coding-triplets and amino-acids.** *Nature. 1965*, **207**:597-599.

(20): Rose GD, Geselowitz AR, Lesser GJ, Lee RH and Zehfus MH: **Hydrophobicity of amino acid residues in globular proteins.** *Science 1985*, **288**:834-838.

(21): Sengupta S, Higgs PG : **A unified model of codon reassignment in alternative genetic codes. Genetics.** 2005, **170**:831-840.

(22): Sengupta S, Yang X and Higgs PG: **The mechanisms of codon reassignments in mitochondrial genetic codes.** *J Mol Evol* 2007, **65**:662-668.

(23): Sonneborn TM : **Degeneracy of the genetic code, extent, nature, and genetic implications. In: Evolvoing genes and proteins.** (Bryson V, Vogel HJ, eds) pp. 377-397. *New York: Academic press.* 1965.

(24): Swire J, Olivia P. Judson, Austin Burt: **Mitochondrial genetic codes evolve to match amino acids requirements of proteins.** *J Mol Evol,* 2005, **60**:128-139.

(25): Weber AL and Miller SL: **Reasons for the occurence of the twenty coded protein amino acids.** *J Mol Evol* 1981, **17**:273-284.

(26): Woes CR: **On the evolution og genetic code.** *Proc Natl Acad Sci USA* 1965, **54**:1546-1552.

(27): Woes CR, Dugre DF, Dugre SA, Kondo M and Saxinger WC: **On the fundamental nature and evolution of the genetic code.** *Cold Spring Harbor Symp Quant Biol* 1966, **31**:723-736.

(28): Wong JT : **A co-evolution theory of the genetic code.** *Proc. Natn. Acad Sci.* U.S.A. 1975, **72**:1909-1912.

(29): Wong JT and Bronskill PM: **Inadequuacy of pre-biotic synthesis as the origin of proteinaceous amino acids.** *J Mol Evol* 1979, **13**:115-125.

(30): Wong JT : **Evolution of the genetic code.** *Microbiol. Sci.* 1988, **5**:174-181.

(31): Zimmerman JM, Eliezer N and Simha R: **The characterization of amino acid sequences in proteins by statistical methods.** *J Theor Biol* 1968, **21**:170-201.

Appendix: A

The code (written in matlab) to calculate p_{ij} :

```
codon_no=zeros(64,3);                    % a null matrix of size 64x3
converter='UCAG';
n=0;
for i=1:4
    for j=1:4
        for k=1:4
            n=n+1;
            codon_no(n, :)=[j i k];
        end
    end
end
codons=converter(codon_no);             % converter converts the numbers in
                                        % U, C, A, G
x=codons;
p=zeros(64,64);                         % null mattix of size 64x64
for i=1:64
    for j=1:64
        y=find(x(i,:)~=x(j,:));         % to check at how many position two
                                        % codons differ
        l=length(y);
        if(l==3)                        % if codons differ at all the three
                                        % positions
            p(i,j)=0;
        end
          if(l==2)                      % if codons differ at any two
                                        % positions
              p(i,j)=0;
          end
            if(l==0)                    %  take zero for all the diagonal
                                        % elements
                p(i,j)=0;
            end
              if(l==1)                  % if there is one base change
```

```
                  if(y==3)                 % if codons are differ at 3rd position.

                      p(i,j)=0.05;         % 0.05 is the value of the parameter
                                            % which control the mistranslation
                                           % errors.
                  end

                       if(y==1)           % if the two codons differ only at 1st position

                           purine_pyrimidine('AGUC')=[1 1 2 2];     % purines
                                             % and pyrimidines are replaced by 1
                                             % and 2 respectively

                           x1=purine_pyrimidine(x);


                           if(x1(i,1)==x1(j,1))    % checks that whether there
                                                 %is transition at the 1st position


                               p(i,j)=0.05;

                           else

                                p(i,j)=0.05/2;

                           end

                       end

                       if(y==2)                   % if the two  codons differ only
                                                   % at the 2nd position
                            purine_pyrimidine('AGUC')=[1 1 2 2];

                            x1=purine_pyrimidine(x);

                            if(x1(i,2)==x1(j,2))   % checks that whether there is
                                                  % transition at the 2nd position.

                                p(i,j)=0.05/2;

                               else

                                   p(i,j)=(0.05)*0.1;
                            end

                       end

               end
    end
end


for i=1:64

    add(i)=sum(p(i,:));

end

for i=1:64




for j=1:64
```

```
        if(i==j)
            p(i,j)=1-add(i);                % fill the diagonal elements
                                            % correctly
        end
    end
end
```

Appendix B

The code (written in matlab) to create a matrix (size 20x64) which gives values of $\varphi i(\alpha)$ and the average cost of the code.

```
D=load ('/home/abhay/Desktop/dwdata');      % 20x20 distance matrix to  give the
                                            %cost of replacement of one amino acid
                                            %by another,here it is  uploaded from the
                                            % Desktop


f1=(F/100)';                           %row vector for the frequency of 20 amino
acids

no_syn_old=[0 0 0 0 16 0 0 0 16 0 0 0 0 0 16 0 0 0 0 16];
                                  % no. of codons coding for amino acids in the 4_column
                                  %code




% mapping between 64 codons and 4 earliest  amino acids(Val,Ala,Asp,Gly) in
%earliest 4_column code  code.

map_old=[5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 15 15
15 15 15 15 15 15 15 15 15 15 15 15 15 15 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20];



 map=map_old;

 no_syn=no_syn_old;

 phi_old=phi(map,no_syn,D);               % to call a function to calculate the
                                         % frequency of codons at each site-type
                                         % which definition is given in section B1.


 phi=phi_old;

 map=map_old;

cost_pre=cost(f1,phi,p,D,map);                 %to call a function to calculate the
                                            % cost which defined in section B2. p is
                                         % the error probability matrix calculated by
                                         % the code given in Appendix A

 fprintf('the cost for four column code is  %f\n', cost_pre);
```

B1:

```
 function phi_value=phi(map,no_syn,D)                    % function definition to
                                                         %calculate the phi.
A=zeros(20,64);

for alpha=1:20                                   % as there are 20 types of the sites

    for i=1:64                                   % there are 64 codons

        A(alpha,i)=D(alpha,map(i));                 % 20x64 matrix; A(i,j) gives the
                                          %distance between the amino acid alpha=1 and
                                           % the amino acid coded by the codon j.
    end
end

for alpha=1:20
    for i=1:64
        [min_val(alpha) pos(alpha)]=min(A(alpha,:));     % ith element of the
                                           % vector min_val and pos gives the  minimum
                                           %element and its position in the ith row of
                                           % matrix A respectively.




        if(map(pos(alpha))==map(i))          % check which codon codes for the the a.a.
                                             % which has minimum distance corresponding
                                              % to the amino acid alpha



            phi_value(alpha,i)=1/no_syn(map(i));         % no_syn(map(i)) gives
                                                 %the no. of synonyms codons for  amino
                                                 %acid coded by codon i.
        else                                   % phi_value(alpha,i) gives of the codon i
                                                 % at the sites of type alpha.
            phi_value(alpha,i)=0;

        end

    end

end

end
```

B2:

```
    function cost_value=cost(f1,phi,p,D,map)   % function to   calculate the
                                                % average cost
cost_value=0;
 for alpha=1:20
     for i=1:64
         for j=1:64
cost_value=cost_value+f1(alpha)*phi(alpha,i)*p(i,j)*D(alpha,map(j));
         end
     end
 end

end
```

Appendix C

```
% for the selection between three codes: one is an original code and other
% two resulted from the original one due to reassignment of some codons to
% other amino acids relative to original code.

 cost_pre=input('enter the cost of the original code>');
 cost_int1=input('enter the intermediate state cost of the deviant code1>');
 cost_int2=input('enter the intermediate state cost of the deviant code2>');
 cp=cost_pre/(cost_pre+cost_int1+cost_int2);
 c1_int=cost_int1/(cost_pre+cost_int1+cost_int2);          % normalisation of
                                                            %the cost values
 c2_int=cost_int2/(cost_pre+cost_int1+cost_int2);
 s=input('enter the selection coefficient');
  wp=1-s*cp;
  w1i=1-s*c1_int;
                                                    % fitness of the  code
  w2i=1-s*c2_int;

 fprintf('the fitness of standard code   %f\n', wp);
 fprintf('the fitness of code1 in intermediate state  %f\n', w1i);
 fprintf('the fitness of code2 in intermediate state  %f\n', w2i);

 n = input('what population size you want >');        % whole population size
 n1 = input('enter the no. of deviant code1 >');
 n2 = input('enter the no. of deviant code2 >');
 n3 = input('enter the no. of original code >');
 t=input('how many times u want to fix the population i.e. no of trials >');
counter1=0;
counter2=0;
counter3=0;

for k=1:t                                            % number of trials  1 for
pop=[ones(1,n1),ones(1,n2)*2,ones(1,n3)*3];         % deviant code1, 2 for code2,
                                                 % and 3 for the original codes

 while(1)
```

```
count=0;
  while(count < n)
           r1=ceil(rand(1)*n);           % generates a random no. Between 1
                                          %and n to slect a individual randomely
           r2=rand(1);                  % generates a random no. between 0 and 1.
           if(pop(r1)==1)                 % check the individual
              if(w1i>r2)
               count=count+1;          % reproduce it with probability proportional
                                         % to its fitness
               temp(count)=1;
              end
           end

           if(pop(r1)==2)
              if(w2i>r2)
                 count=count+1;
                 temp(count)=2;
               end
           end


            if(pop(r1)==3)
             if(wp>r2)
               count=count+1;
               temp(count)=3;
             end
           end
     end
        pop=temp;
        n_code1=0;
        n_code2=0;
        n_code3=0;

for i=1:n                         % count total no. of each types after each generation
   if(temp(i)==1)
      n_code1=n_code1+1;
   end
   if(temp(i)==2)
```

```
            n_code2=n_code2+1;
        end
        if(temp(i)==3)
            n_code3=n_code3+1;
        end


    end
        if(n_code1==n)                      % check which code has been fixed
            counter1=counter1+1;            % counter 1 will give how many times
                                             %  code1 has been fixed.
           Break;
        end
        if(n_code2==n)
            counter2=counter2+1;                  % counter2 will give how many times
                                            %code2 has been fixed.
           Break;
        end



if(n_code3==n)                              % counter3 will give how many times i.e.
                                           % original code has been fixed out of t trials
      counter3=counter3+1;
          break;
       end
     end
end

 fprintf('The no by which code1 has been fixed is  %d\n', counter1);
 fprintf('The no by which code2 has been fixed is  %d\n', counter2);
 fprintf('The no by which code3 ie. standard code has been fixed is  %d\n',
 counter3);

 fix_prob1=counter1/(counter1+counter2+counter3);  % fixation probability of
                                                     % code1
 fix_prob2=counter2/(counter1+counter2+counter3);  % fixation probability of
```

```
                                                     %code2
fix_prob3=counter3/(counter1+counter2+counter3);  % fixation probability  of the
                                                   % original code


fprintf('the fixation probability of the deviant code1 is  %d\n', fix_prob1);
fprintf('the fixation probability of the deviant code2 is  %d\n', fix_prob2);
fprintf('the fixation probability of the original code is  %d\n', fix_prob3);
```

Appendix D

```
 cost_pre=input('enter the cost of the  code1');
  cost_int1=input('enter the cost of the  code2 ');
  cost_int2=input('enter the cost of the code3');
  c1_int=cost_pre/(cost_pre+cost_int1+cost_int2);      % normalisation of the cost
                                                        % values
  c2_int=cost_int1/(cost_pre+cost_int1+cost_int2);
  c3_int=cost_int2/(cost_pre+cost_int1+cost_int2);

  s=input('enter the selection coefficient');
  w1i=1-s*c1_int;                                     % fitness of the original code
  w2i=1-s*c2_int;                                     % fitness of the deviant code1
  w3i=1-s*c3_int;                                     % fitness of the deviant code2

  fprintf('the fitness of origi  %f\n', w1i);
  fprintf('the fitness of code2  %f\n', w2i);
  fprintf('the fitness of code 3 %f\n', w3i);

 display( 'THE FIXATION BY WITHOUT MORAN PROCESS');
n = input('what population size you want >');
n1 = input('enter the no. of  code1 >');
n2 = input('enter the no. of code2 >');
n3 = input('enter the no. of code3 >');


display('enter the value of that flag as 1 if u want to stop the simulation
when that code is fixed and other flags as 0');
display('if we set flag1=1 and other flag values as 0,then simulation stop
when code1 will get fixed');
flag1=input('enter the value of flag1>');
flag2=input('enter the value of flag2>');  % if we have to do the simulation
                                            % untill code 3 gets fixed then set
                                           %flag1=0, flag2=0 and flag3=1
flag3=input('enter the value of flag3>');
flag=0;
trial=0;
```

```
num=0;
while(flag==0)                      % when the desired code will be fixed, flag will
                                     % take value 1 and simulation will be stopped.
 trial=trial+1;
 pop=[ones(1,n1),ones(1,n2)*2,ones(1,n3)*3];
 temp=zeros(1,n);
 code1=0;
 code2=0;
 code3=0;
 num=0;
 while(1)
     num=num+1;
     count=0;
           while(count < n)
                r1=ceil(rand(1)*n);       % generate a random no. between 1 and n.
                r2=rand(1);                  % generate a random no. between 0 and 1
                  if(pop(r1)==1)
                     if(w1i>r2)
                      count=count+1;
                      temp(count)=1;
                     end
                 end
                if(pop(r1)==2)
                   if(w2i>r2)
                    count=count+1;
                    temp(count)=2;
                   end
                end
                if(pop(r1)==3)
                   if(w3i>r2)
                    count=count+1;
                    temp(count)=3;
                   end
                end
```

```
   end
     pop=temp;

     n_code1=0;

     n_code2=0;

     n_code3=0;

  for i=1:n                                   % after each generation count
                                             % the different types individuals
    if(temp(i)==1)

      n_code1=n_code1+1;

    end

  end


  for i=1:n

    if(temp(i)==2)

      n_code2=n_code2+1;

    end

  end


  for i=1:n

    if(temp(i)==3)

      n_code3=n_code3+1;

    end

  end


code1(num)=n_code1;

code2(num)=n_code2;

code3(num)=n_code3;



 if(n_code1==n)

    flag=flag1;

    break;

 end

 if(n_code2==n)      % when the disered code is fixed flag will take value 1.
```

```
          flag=flag2;
          break;
       end

        if(n_code3==n)
          flag=flag3;
          break;
        end


 end
end
 display('YOU CAN SEE THE VARIATION OF POPULATION>');
 display('TYPE code1,code2,code3 RESPECTIVELY TO SEE THE VARIATION OF
POPULATION WITH TIME');

 % code 1 is an array that gives the no. of code1 in each generation.
 % code 2 is an array that gives the no. of code2 in each generation.
 % code 3 is an array that gives the no. of code3 in each generation.
```