

An Application of Artificial Intelligence and Machine Vision to
Protein Engineering

by

Adam Paul Arkin
B.A. (Honors) in Chemistry, Carleton College (1988)

SUBMITTED TO THE DEPARTMENT OF CHEMISTRY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY IN CHEMISTRY

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY
June, 1992

©Massachusetts Institute of Technology
All rights reserved

Signature of Author **Signature redacted**
Department of Chemistry
May 20, 1992

Certified by **Signature redacted**
Keith A. Nelson
Thesis Supervisor

Certified by **Signature redacted**
Douglas C. Youvan
Thesis Supervisor

Accepted by **Signature redacted**
Glenn A. Berchtold, Chairman,
Departmental Committee on Graduate Students
Department of Chemistry

ARCHIVES

MASSACHUSETTS INSTITUTE
OF TECHNOLOGY

JUL 30 1992

LIBRARIES

This Doctoral thesis has been examined by a Committee of the Department of Chemistry as follows:

Professor Robert J. Silbey Signature redacted
Chairman

Professor Keith A. Nelson Signature redacted
Thesis Supervisor

Professor Douglas C. Youvan Signature redacted
Thesis Supervisor

Professor Stephen J. Lippard Signature redacted
Committee Member

Professor William H. Orme-Johnson Signature redacted
Committee Member

An Application of Artificial Intelligence and Machine Vision to Protein Engineering

by Adam Paul Arkin

Submitted to the Department of Chemistry on May 21, 1992 in partial fulfillment of the requirements for the Degree of Doctor of Philosophy in Chemistry

Abstract

Protein engineering is defined as the design and production of proteins which exhibit specified behaviors. Despite many advances, both in theory and experiment, it is still impossible to predict the structure and function of a protein given its primary sequence of amino acids. The inverse problem, predicting the primary sequence of amino acids which will produce a given structure or function, is almost certainly even more difficult. This thesis presents two new technologies to help bypass the need to solve the inverse protein folding problem. The first is an experimental mutagenesis protocol, Recursive Ensemble Mutagenesis, which performs a highly parallel adaptive search of sequence space for proteins fitting a design criterion. The second technology is Digital Imaging Spectroscopy which is able to obtain the spectrum from every point in a two-dimensional image. The spectrometer is used to analyze the *in vivo* ground-state absorption spectra of the photosynthetic proteins of the bacterium *Rhodobacter capsulatus*.

Sequence space is defined as the set of all possible proteins. If we only consider proteins of 1000 amino acids or less, then there are greater than $20^{1,000}$ proteins contained by this space (more particles than there are estimated to be in the universe!) Presumably, each of these sequences is able to perform a given function to a greater or lesser extent determined by some highly non-linear relation. Searching such a space for the protein which best fits a desired criterion can be shown to be at least NP-hard, i.e., the optimum cannot, on average, be found in a reasonable amount of time. Therefore, it is necessary to employ approximate combinatorial optimization algorithms to search the space effectively. Recursive Ensemble Mutagenesis (REM) is such an algorithm. Similar in many respects to the artificial intelligence

technique of genetic algorithms, REM iteratively "evolves" a diverse population of trial solutions to fit a specific design criterion.

REM employs combinatorial cassette mutagenesis to create a randomized population of mutant proteins called an *ensemble*. The ensemble is subjected to a selection or screen which identifies mutants displaying desirable properties. The sequences (either DNA or amino acid) of these "fit" mutants are analyzed and used to calculate the nucleotide composition of a new combinatorial cassette. A new ensemble is created by the introduction of this cassette into the gene for a protein and the process begins again. Simulations of this technique indicate that REM is able to produce a large diverse set of proteins fitting an engineer's specifications while examining only a miniscule portion of the sequence space available to it. Methods for the calculation of a combinatorial cassette from an ensemble of proteins are discussed along with the the relevant experimental limitations.

One of the most important experimental limitations on REM, or any combinatorial cassette experiment, is the maximum number of mutants which can be screened in a reasonable amount of time. The greater the proportion of the ensemble which can be examined, the more information about the constraints on protein function can be gleaned. The Digital Imaging Spectrometer (DIS) was developed in anticipation of screening combinatorial libraries of mutants in the photosynthetic apparatus of *Rb. capsulatus*. The VIS/NIR absorption spectra of these membrane proteins are a sensitive indicator of structure and function. Isolating the proteins in a form suitable for a conventional spectrometer is a time consuming process, taking more than a week in some cases. However, using specialized machine vision techniques, DIS is able to obtain analytical ground-state absorption spectra from more than 1,000 bacterial colonies simultaneously in under twenty minutes. Spectra are recorded at a maximum of 5nm resolution over the wavelength range from 930-400nm. Noise in the spectra is on the order of 0.0005 OD to 0.005 OD at low and high absorption extremes respectively. A suite of algorithms is used to classify and display the large number of resulting spectra.

Thesis Supervisor: Keith A. Nelson

Title: Professor of Chemistry

Thesis Supervisor: Douglas C. Youvan

Title: Associate Professor of Chemistry

Acknowledgements

Doug hates these things but I'm going to do one anyway! (So there!) First thanks goes, of course, to my mentor Douglas Youvan who provided much of the original thought which went into the production of this thesis. I want to thank him for providing an environment where controlled insanity could actually lead to scientific progress and for being a great sparring partner and friend.

Thanks also goes to Keith Nelson, who first connected me to Dr. Youvan and then allowed me to pursue the rather odd path described in this thesis. I thank Keith for his enthusiasm and support whenever we spoke.

I would also like to thank the Youvan Clan: Ellen, Steve, Bill, Mary, Simon, and Christine. Ellen was my fellow traveller, a wayward waif drawn from the fold of the '88 pchem subsection into the realm of biological goop. Thanks go to her for walking this confusing trail with me. Bill was the Snark Master. Thanks go to him for many interesting discussions and, of course, meals. (Is it lunch time yet?) Thanks also to Mary, Steve and, yes, Simon for providing both reality checks and good humor. Finally, I want to thank Christine for letting me tease her so unmercifully. I wouldn't have done it if I didn't like her and respect her so much.

Special thanks goes to David Chasman. He is one of the only true scholars I have ever met. I wish to thank him for the many lengthy (and sometimes loud) argument over subjects which even a theorist would call mental masturbation.

There are so many other people who were fundamental in giving me confidence in myself, love and support: there is Thea who is my greatest friend despite our distances, Carolyn who was my companion for four years and who is the sweetest person I have ever met. There is Dan Sodickson, James Myers, Steve Dinneen, Nat Case, Don Owen, Will Cramer and Julie Thomas who provided an excellent community of friends during my years of college and/or graduate school. Special thanks to the latter two who were actually able to live with me for over three years. I will miss our weekly pizza and Star Trek dinners. Finally, thanks go to Aline Cornelius who has been my pal since my start at M.I.T. and has slowly become one of my closest friends. (Her fault!)

Finally, grateful thanks to my family for...well....being my family. I can't think of another one to which I would like to belong. It is to them that this thesis is dedicated.

1.1 Adult in Progress Series 1

1.2 In Search of a Guide 3

1.3 Growth & Impulsivity 3

1.4 Social and Cultural Influences 4

1.5 Personality & Emotional Maturity 12

1.6 Summary 15

Chapter 2: Digital Imaging Spectroscopy 17

2.1 The Evolution of a Digital Imaging Spectroscopy 19

2.2 Digital Imaging Spectroscopy 21

2.2.1 Summary 21

2.2.2 Applications for Library Research 23

2.2.3 Comparison with Special Types of Data 25

2.2.4 FIS: A Library Specific Model 27

2.2.4.1 FIS Layers 27

2.2.4.2 FIS Light Source and Interactions 33

2.2.5 FIS Software Specifications 35

2.2.5.1 FIS Analysis 36

2.2.5.2 FIS Data Acquisition 45

2.2.5.3 FIS Data Archiving and Display 49

2.2.5.4 FIS User Level Programs 52

2.2.6 Characteristics 60

2.2.7 Applications 61

2.2.8 Summary 61

2.3 FIS in the Library 62

2.4 Summary 63

Chapter 3: Optimized Information Systems 64

3.1 Reducing Complexity 70

3.2 Optimized Reference Models for Continuous Current Multiple 70

3.2.1 Summary 70

3.2.1.1 OCM or Search Request Spectrum 71

3.2.2 Results and Discussion 73

3.2.3 Acknowledgments 81

3.3 Epilogue 84

Table of Contents	Page
Chapter 1: Combinatorics in Sequence Space	1
1.1 Adrift in Sequence Space	2
1.2 In Search of Proteins	
Genetic Manipulations	3
1.3 Combinatorial Optimization	8
1.4 Recursive Ensemble Mutagenesis	12
1.5 References	13
Chapter 2: Digital Imaging Spectroscopy	18
2.1 The Evolution of a Digital Imaging Spectrometer	19
2.2 Digital Imaging Spectroscopy	23
2.2.1 Summary	23
2.2.2 Machine Vision for Library Screening	23
2.2.3 Construction of a Spectral "Tester" Strain	25
2.2.4 DIS Hardware Specifications	28
2.2.4.1 The CCD Camera	30
2.2.4.2 The Light Source and Integrating Sphere	33
2.2.5 DIS Software Specifications	35
2.2.5.1 Image Analysis	36
2.2.5.2 Spectral Acquisition	45
2.2.5.3 Classification and Display	48
2.2.6 System Development	59
2.2.7 Conclusions	60
2.2.8 Acknowledgements	61
2.3 Epilogue	61
2.4 References	66
Chapter 3: Optimized Nucleotide Mixtures	69
3.1 Reducing Complexity	70
3.2 Optimized Nucleotide Mixtures for Combinatorial Cassette Mutagenesis	70
3.2.1 Summary	70
3.2.1 CCM to Search Sequence Space	71
3.2.2 Results and Discussion	75
3.2.3 Acknowledgement	84
3.3 Epilogue	84

3.4 References.....	88
Chapter 4: Recursive Ensemble Mutagenesis	92
4.1 Forward.....	93
4.2 Simulations of Recursive Ensemble Mutagenesis.....	93
4.2.1 Summary.....	93
4.2.2 Reverse Engineering the Protein Folding Problem.....	93
4.2.3 Description and Results of Simulations.....	97
4.2.3.1 Theory and Description of Algorithm.....	97
4.2.3.2 General Aspects of REM Simulations.....	101
4.2.3.3 REM Simulations Using an "Alphabetic" Decision Algorithm.....	105
4.2.3.4 REM Simulations on an Hydropathically Constrained Domain.....	108
4.2.3.5 REM Simulation of a Complex Binding Site.....	108
4.2.3.5 REM Simulations Involving an Alpha Helix.....	110
4.2.3.6 Acceleration of REM.....	111
4.2.4 Conclusions The Relation of REM to GAs, etc.	112
4.2.5 Acknowledgement.....	114
4.3 Epilogue.....	114
4.4 References.....	118
Chapter 5: The Effects of DA Stringency and Library Complexity on REM	122
5.1 Forward	
Experimental Limitations on REM.....	123
5.2 The Effects of DA Stringency and Library Complexity on REM	124
5.2.1 Summary	124
5.2.2 Complexity and Stringency.....	124
5.2.3 Mathematical Background.....	125
5.2.4 Simulations	128
5.2.5 Acknowledgments	134
5.3 Epilogue.....	134
5.4 References.....	137

Chapter 6: Towards a Practical Implementation of REM.....	140
6.1 Forward.....	141
6.2 REM Revisited.....	141
6.2.1 Summary.....	141
6.2.2 Introduction.....	142
6.2.3 Experimental Aspects of REM.....	144
6.2.4 Overview of REM Methodology.....	145
6.2.5 Computer Simulations Using REM.....	147
6.2.6 SSD versus pG.....	152
6.2.6 Acknowledgments.....	155
6.3 Epilogue.....	155
6.4 References.....	159
Appendix A: A Bibliography of Programs.....	162
A.1 Post-Mortem.....	163
A.2 Annotated List of Programs.....	163

Chapter 1

1.1 Adrift in Sequence Space

Combinatorics in Sequence Space

The goals of a protein engineer are manifold, ranging from the design of enzymes and antibodies to the development of vaccines and the identification of drug targets. In all cases, the engineer is faced with the task of navigating a vast, complex sequence space. This space is defined by the possible combinations of amino acids at each position in a protein sequence. For a protein of length n , the total number of possible sequences is 20^n , where 20 is the number of different amino acids. This is a very large number, even for a relatively small protein. For example, a protein of length 100 has 20^{100} possible sequences, which is approximately 1.27×10^{130} . This is a number that is far larger than the number of atoms in the observable universe. This vastness of sequence space is what makes protein engineering such a challenging task. It is often compared to finding a needle in a haystack, or to finding a specific path through a dense forest. The engineer must be able to navigate this space efficiently, identifying promising candidates for further study. This is where combinatorics comes in. Combinatorics is the branch of mathematics that deals with the counting of objects, and it provides a powerful framework for understanding the structure and complexity of sequence space. In this chapter, we will explore the basics of combinatorics, and how it can be applied to the design of proteins and other biological molecules. We will start by discussing the fundamental principles of counting, and then move on to more advanced topics such as permutations, combinations, and the binomial theorem. We will also see how these concepts can be used to calculate the number of possible sequences for a given protein, and to estimate the probability of finding a specific sequence. Finally, we will discuss some of the practical applications of combinatorics in protein engineering, such as the design of libraries of random sequences and the optimization of protein sequences. By the end of this chapter, you should have a solid understanding of the combinatorial aspects of sequence space, and be able to apply these concepts to your own work in protein engineering.

Naturally occurring proteins have evolved through a process of spontaneous (non-Lamarckian) genetic mutation. However, mutants are judged to have an advantage only if they are better at surviving and reproducing in their environment.

1.1 Adrift in Sequence Space

The goals of a protein engineer are manifold, ranging from the design of pharmaceutically active peptides and catalysts for organic reactions to the production of organisms constructed for bioremediation. However, given the current state of theory it is as yet impossible to predict the function and three-dimensional structure of a protein from its primary sequence of amino acids. Even, if this were possible, the inverse problem would have to be solved: how does one predict the protein sequence which best produces a given biological function. The search for this postulated protein constitutes a traversal of sequence space. Sequence space has been envisioned as a discrete surface whose defining axes enumerate the amino acids which may appear at each site in a protein's primary sequence.¹ For example, for proteins of length 1,000, the space is a 1,000 dimensional hypercube wherein each of the $20^{1,000}$ possible proteins is represented by a single vertex. The "height" of each point in this space indicates the *fitness* of the protein for the performance of a specified task. Since, for example, single point mutations in a protein sequence may destroy its functionality the topography of sequence space is assumed to be very rugged, composed of steeply defined ridges, valleys, and plateaus.² Thus, knowledge of the fitness of a specific protein sequence will not, in general, give any information on the fitness of its "nearest" neighbors. It has been shown that for any high-dimensional parameter space with a highly non-linear objective function, the search for a global optimum is, at best, NP-hard.³ This implies that standard optimization procedures, such as gradient algorithms, will almost certainly fail to consistently locate fit proteins in sequence space in a reasonable amount of time and a combinatorial, stochastic approach is required.

Naturally occurring proteins have evolved according to spontaneous (non-Larmarkian) genetic mutation.⁴ Random mutants are judged fit or unfit by the environment in which they were

generated. By virtue of the large number of organisms undergoing these random perturbations and selections, it can be argued that Nature has been employing combinatorial optimization techniques for millions of years. In fact, by 1930 the mathematics of Mendelian genetics had advanced far enough to prove that, given a geological time-scale and at least a mild-selection, the very small observed rate of spontaneous mutation of genes ($<1 \times 10^{-7}$) was enough to explain the accumulation of advantageous phenotypes.⁵⁻⁷ Further, since the turn of the century it had been recognized that the study of these spontaneously occurring mutations could yield valuable insight into biochemical processes. For example, as early as 1908, observations of hereditary diseases in humans lead Garrod⁸ to propose the one gene-one enzyme hypothesis, a proposal which motivated Beadle and Tatum^{9, 10}, in 1940, to attempt to increase the rate of mutation in *Neurospora*, using UV light, in order to isolate growth factor mutants and identify the enzymes responsible for the metabolic deficiency. It thus became obvious that mutagenesis, the production of *man-made* mutations, would be an indispensable tool for understanding the mechanisms of life.

The balance of this chapter is dedicated to an abbreviated history of the evolution of mutagenesis up until the time of the inception of this thesis and a discussion of how the relatively new methods of combinatorial optimization may be used to make the necessary stochastic search of sequence space more efficient.

1.2 In Search of Proteins: Genetic Manipulations

Perhaps the earliest mutagenesis experiments were performed after the observations by Muller¹¹ and Stadler¹², in 1927, that X-rays increase the rate of spontaneous mutation. Since large genes are more likely to be hit by an X-ray, these genes are commensurately more susceptible to mutation. Thus, X-ray mutagenesis was used to provide a rough estimate of gene size. X-rays are non-specific and induce random mutations throughout the entire genome. This type of random mutagenesis, wherein the locus

and type of mutation are (initially) unknown, is representative of the type of work which ensued for the next forty years. Much of the biochemistry of metabolic cycles and reproduction was elucidated by the combination of these mutagenesis experiments with standard genetic crossing experiments. However, if the details of these processes were to be investigated, finer control of the genetic regulatory elements and proteins would be required.

The discovery of restriction enzymes in 1970 opened the doorway towards this more exquisite control.^{13, 14} Class II restriction enzymes recognize specific palindromic DNA sequences (typically six bases in length) and cut at those locations. In combination with the ability to utilize bacterial plasmids and phage, these restriction enzymes allowed for the production of recombinant organisms and, notably, a simple method for DNA sequencing. Once the genes responsible for a given function could be isolated on small, manipulable genetic elements it was plausible to think of targeting those specific genes for mutagenesis. In fact, the most obvious way to do this (in hindsight) is to chemically synthesize the part of the gene one wished to mutagenize. The earliest experiments involved isolating a gene of interest on a plasmid, cutting it with a restriction enzyme at one site, removing nucleotide bases from the end using nucleases then recircularizing the plasmid using a chemically synthesized linker fragment and the enzyme ligase. It wasn't until the late seventies that the technology existed for the synthesis of entire genes and a method for rapid site-directed mutagenesis was proposed.

Up until about 1978, site-directed mutagenesis was carried out using chemicals which interacted with individual nucleotides. For example, hydroxylamine reacts solely with cytosine, hydroxylating it such that it may only base-pair with adenine. This effects a GC to AT base-pair transition in the gene. Knowledge of the gene sequence would allow the prediction of which sites would be mutagenized. When the ability to chemically synthesize fragment of DNA became feasible, another method presented itself,

oligonucleotide mediated mutagenesis.¹⁵ When a small oligomer of nucleotides is synthesized such that it is complementary to a target region of the gene, it can then be expected to anneal specifically to that section under the proper conditions. If a small number of bases internal to the oligonucleotide are *not* complementary to those naturally occurring in the gene section, the oligomer will still anneal to the original locus albeit less stably. When the rest of the gene is replicated from this small duplex, one half of the resulting genes will contain complementary mutations at the mismatched sites. This process is made very efficient by the use of the filamentous *E. coli* phage M13 which expresses a single-stranded form of its genome and by a technique, developed by Kunkel¹⁶, employing mutant *E. coli* strains. Use of site-directed techniques and methods relying on synthesis of sections of genes finally allowed proteins to be engineered and produced in a rational fashion.

At this point the problems discussed in the introduction became apparent. As a functional probe, mutagenesis is very useful, but it is highly nontrivial to predict the effect a given mutation will have in an organism or to design a protein to exhibit a specific behavior. To circumvent this deficiency one could only try all possible mutations at a given site and analyze each mutant. This was accomplished by site-saturation mutagenesis wherein populations of oligonucleotides were synthesized such that at certain positions there was an equal chance that any of the four bases would be incorporated.¹⁷ However, single site-saturation experiments do not admit to compensatory effects which occur when more than one site is mutated at once. For example, there are many instances in which a single amino acid change in a protein will restore activity to an inactive mutant enzyme.¹⁸ Worse, compensatory mutations may occur anywhere in the protein relative to the original inactivating mutation. It was an obvious next step to randomly mutate more than one site at a time. This was first accomplished by Matteucci and Heyneker¹⁹ who explored the effects of randomizing a nine-base pair region immediately 5' to the bovine growth hormone structural gene initiation site. This was achieved by synthesizing a random

population of DNA duplexes (cassettes) with "sticky" ends which may be ligated into DNA which has been cut with restriction enzymes. This technique is called Combinatorial Cassette Mutagenesis (CCM) and has since been used for a variety of tasks such as exploring effects of hydrophobic packing in the internal core of globular proteins^{20, 21} and discovering the optimal sequence for a ribosome binding site.²² Related targeted gene-randomizing methods have also become popular and have had success, for example, in increasing the catalytic efficiency of triose-phosphate isomerase²³ and producing temperature sensitive variants of a cytochrome *c*.²⁴

The largest problems with CCM are conservation of the library complexity through many experimental steps (DNA synthesis, ligation into a vector, transformation of the organism, etc.) and the exhaustive screening of the library once it is expressed. Thus, CCM is only practical when the screening procedure is capable of identifying and isolating a large number of the resultant genes. In some cases, a lethal selection or a rapid *in vivo* phenotypic screen may be employed (see Chapter 2), but in many instances, the gene or gene-product must be individually isolated and examined. The advent of phage-display libraries may ameliorate this problem.

Phage-display libraries were first reported in 1985 by George P. Smith.²⁵ In his work, a random library of DNA fragments was inserted into Gene III, which codes for a small coat protein of the filamentous phage f1. The resultant random peptide library appeared on the coat of the phage in immunologically active form without significant disruption of native phage activity. The phage library was affinity purified by exposure to an antibody and amplified more than 1,000-fold over normal (wild-type) phage. The important results are that insertion of a foreign peptide on the phage coat did not disrupt activity, a large library of peptides could be screened for activity, and successful members of the population could be amplified by simple transfection and replication. Since 1985, phage-fusion technology has become very efficient, allowing the expression and screening of greater than 10^9 clones, and has

extended to the fusion of whole, multi-subunit proteins into the phage coat.²⁶ It is not clear, however, what the upper limit on the size of the fusion protein might be (it is limited to some degree, for example, by the size of the F-pili). Further, even if a fusion protein has a high affinity for a ligand, it is not obvious that it will maintain its affinity in free form or that it will then exhibit catalytic activity. Finally, the phage-display method may well be inapplicable to membrane-bound proteins and proteins whose activity does not involve binding a substrate (e.g. electron transfer proteins).

It should be noted that when a combinatorial library is used to produce mutant proteins the structure of the genetic code must be taken into account. The genetic code is degenerate and thus complete randomization of a codon results in inequivalent proportions of the twenty-amino acids and a stop command. By choosing the proportions of the nucleotides which are used at each codon position, one may try and minimize the appearance of stop codons and choose, to some degree, the relative proportions of the amino acids. Traditionally, only the base composition of the third codon position is limited in an attempt to increase the equiprobability of the amino acids and express only one of the stop codons.^{27, 28} Recently, libraries which express limited sets of amino acids at a site have also been designed.²⁹ Chapter 3 discusses a mathematical method for producing such libraries.³⁰

Another approach to production of combinatorial libraries bypasses genetic manipulation altogether. In 1984, before combinatorial mutagenesis was fully established, Geysen *et al.* described a method for direct chemical synthesis of peptides on a solid support.³¹ These peptides were used to find the epitopes for a given antigen. This allowed for the identification of the antigenic regions of biologically important proteins by simple sequence comparison. The chemical approach was recently improved by Fodor *et al.*³² using a light-mediated lithographic process and Lam *et al.* using chemically produced peptide bead libraries.³³

An interesting technique was developed from this technology, first by Geysen *et al.*³⁴ and then by Houghten *et al.*^{35, 36}, whose logic may have application to the mutagenesis techniques described above and in Chapters 4 and 6. These researchers constructed peptide libraries to probe immunological affinities. In Houghten's work (1991), for example, a six amino acid peptide was designed to inhibit the interaction between a catalytic antibody and a specific 13-residue peptide. To reduce complexity, 324 libraries were synthesized each of which contained a different pair chosen from a set of 18 amino acids (tryptophan and cysteine were omitted from the synthesis) in the first two positions of the peptide and a random sequence in the final four positions. The library demonstrating the greatest overall affinity (in this case a library with DV (single letter code) initiating the peptide) is used to derive twenty more libraries which conserve the first two peptide positions, set the third position to one of the twenty amino acids, and randomize the remaining three. The process is repeated until there are no more sites to randomize. In this way, a peptide sequence with a very high inhibition is obtained. This step-through method may, theoretically, be applied to any CCM experiment by stepping through the gene in pieces.

1.3 Combinatorial Optimization

All the above combinatorial methods have an upper limit to the complexity they are able to produce and screen. This number is usually much less than the size of the total sequence space available around a given protein and thus it is possible that many optima, including global optima, in the space will not be observed. If we are interested in finding optima, then it would be advantageous to be able to use information gained from examining a small subset of the sequence space to direct the search into a more optimal region. If the space is completely random, then this obviously cannot be done and all possible sequences will have to be enumerated and tested, a problem which grows exponentially with the length of the sequence.

In any case, as mentioned above, optimization in any high-dimensional space with non-linear constraints can be shown to be NP-hard.³ That is, the problem can only be solved in greater than polynomial time, a constraint which makes exact algorithms intractable. Since the function which determines optimality in sequence space is, in general, quite non-linear, there is little hope of solving the inverse protein folding problem exactly. Thus, approximate algorithms must be employed.

Two very general and popular approximate combinatorial-optimization procedures are simulated annealing³⁷ and genetic algorithms.³⁸ Interestingly, both these methods derive from analogies to natural physical processes. The use of annealing in combinatorial optimization was introduced independently by Kirkpatrick, Gelatt & Vecchi^{39, 40} and Cerny⁴¹ during the early 1980's. Simulated annealing begins with a trial solution to a problem. This trial solution, S , is randomly modified by a relatively small amount and if the resultant guess, S^* , is more optimal than the original it is accepted, otherwise it is only accepted based on a Boltzmann factor. This factor is of the form $\exp(\beta(E(S)-E(S^*)))$ where β is, metaphorically, an inverse temperature and $E(S)$ is the function to be optimized (typically, minimized) evaluated with the trial solution, S . If a random number is less than this factor, the new solution is accepted otherwise it is rejected. As β tends towards infinity it becomes less likely that an "up-hill" modification will be accepted and when β nears infinity a simple descent algorithm is obtained. The algorithm is initialized with small β (high temperature), and over many iterations the trial solution is modified according to the above protocol as β is gradually increased. This is analogous to cooling an initially hot substance until it reaches a frozen configuration. If this process is performed slowly enough, the final configuration should be the one lowest in energy. Otherwise, disorder will be "frozen in" and a sub-optimal solution will be obtained. Metaphorically, this is what occurs to the solution of the optimization function. Simulated Annealing has gained much support because of its simplicity and

because analytic theories allow the derivation of optimal "annealing" schedules and have proven that the annealing processes are ergodic (a property related to the ability to sample the problem space efficiently).³⁷ Despite it's popularity, there does not seem to be any feasible way to integrate this methodology into an experimental search of sequence space. A more naturally adaptable algorithm, and one which may be proven to have many of the same convergence properties of simulated annealing⁴² is called a Genetic Algorithm.

By 1960, a substantial bestiary of mutagenic mechanisms was known. These included, spontaneous single-base mutation, cross-over, deletion, insertion, inversion, etc., all caused by different malfunctions in the cellular machinery of replication. Inspired by arguments that these types of mutations were somehow optimal for producing fit organisms over many evolutionary generations, John Holland, who had been working in the theoretical computation field of adaptive systems theory, designed an optimization procedure which came to be known as the genetic algorithm (GA).⁴³⁻⁴⁵ In this algorithm an initial population of randomly generated trial solutions (chromosomes) is constructed. Each member of the population is passed to the optimization function and assigned a "fitness". For example, if the function were to be maximized then the fitness could be the value of the function evaluated at the trial solution. Once fitnesses are assigned, pairs of trial solutions (parents) are chosen and "mated" to produce members of the next population (children). The parents are chosen stochastically such that those with higher fitness are more often selected. The two classic mating operators for GAs are analogies for genetic processes, "cross-over" and "mutation". To demonstrate these procedures, assume that a trial solution is represented as an l -bit binary number. The simple cross-over operator randomly chooses a number, n , between 1 and $l-1$ then creates children by concatenating the first n bits from one parent and the second $l-n$ bits from the second parent and *vice versa* (to produce two children). This process is shown below for $n=3$ and $l=9$:

Parent 1: 100100|110

Parent 2: 011101|101

Child 1: 100100101

Child 2: 011101110

The rate of cross-over is generally greater than 0.5. The mutation operator merely flip bits at a low rate, typically 10^{-3} . Many other types of genetic operators, such as n-point and uniform cross-over, have also been used with success. Once a specified number of children are produced they replace the lowest fitness parents. The new list of trial solutions are then passed to the optimization function and the process is repeated. Holland proved that schema (patterns of bits or whatever defines the form of the trial solution) which appear frequently in chromosomes of above average fitness in a given generation are exponentially more frequent in subsequent generations. Further, he proved that $O(N^3)$ (read: order N-cubed) schema are usefully processed by a GA operating on a population containing N members. This lead to the concept of *implicit parallelism* which was used to explain the efficiency of the GA at finding optima. The underlying hypothesis of genetic algorithms is that the cross-over operation has a high chance of producing fitter children. This is certainly the case when there is linear independence between sites in the chromosome, but when there is a degree of interaction it becomes harder to prove. In fact, there is much theory as to what types of interaction make a problem GA-hard.^{46, 47} Obviously, similar factors are responsible for the sub-optimality of other algorithms. Anecdotally, however, GAs have been very successful with many non-linear problems.⁴⁸⁻⁵⁰

Since GAs are based on a genetic metaphor it is easy to see how they might be applied to genetic problems. However, to date, they have never been applied to such problems. Chapters 4-6 describe an algorithm for the experimental search of sequence space for proteins fitting a specified criterion, Recursive Ensemble Mutagenesis (REM). Though, REM was designed independently of GAs,

it bears many similarities to these algorithms (as well as significant differences). Therefore, many of the parameters which are important to the efficiency of GAs will be important to REM. Similarly, the same problems which afflict the functioning of GAs will afflict REM as well.

1.4 Recursive Ensemble Mutagenesis

In this discussion and in most instances, REM is assumed to be employed in the engineering of a protein. REM, however, may be used to operate upon any genetic element whose functionality may be screened efficiently (e.g. a ribosome binding site or a promoter) REM is initiated by generating a random population of DNA using combinatorial cassette mutagenesis. The DNA is passed into an organism and a functional selection or screen is performed. For example, if a β -lactamase is randomly mutagenized, the selection may be survival on a β -lactam antibiotic plate. Those members of the population which pass the screening are sequenced and a new combinatorial cassette is constructed based on the analysis of the resulting sequences (see Chapters 3,4 and 6). This cassette is used to produce the "child" population and this population undergoes another screening. Iterating this procedure results in a diverse set of DNA (or proteins) satisfying an engineer's specifications. All that is required is an efficient method of producing libraries and screening them. REM essentially attempts to evolve a solution to the inverse protein folding problem by a highly parallel, adaptive walk through sequence space. A discussion of differences between REM and GAs and a computer based investigation of its detailed behavior is presented in Chapters 4 and 5. Chapter 6 describes a modification of REM such that it may be used with current DNA synthesizer technology.

1.5 References

1. Smith, J. M. 1970. Natural Selection and the Concept of a Protein Space. *Nature* **225**: 563-564.
2. Palmer, R. 1989. Optimization on Rugged Landscapes. *In* Workshop on Applied Molecular Evolution and Maturation of the Immune Response. A. S. Perelson and S. A. Kaufman. (Ed.) 3-25.
3. Torn, A. and A. Zilinskas. 1989. Global Optimization. Springer-Verlag, Berlin.
4. Luria, S. E. and M. Delbruck. 1943. Mutations of Bacteria from Virus Sensitivity to Virus Resistance. *Genetics* **28**: 491-511.
5. Wright, S. 1931. Evolution in Mendelian Populations. *Genetics* **16**: 97-159.
6. Fisher, R. A. 1930. The Genetical Theory of Natural Selection. Clarendon Press, Oxford.
7. Haldane, J. B. S. 1932. The Courses of Evolution. Harper and Row, New York.
8. Garrod, A. E. 1908. Inborn Errors of Metabolism. *Lancet* **2**: 1-7, 73-79, 142-148, 214-220.
9. Beadle, G. W. and E. L. Tatum. 1941. Genetic Control of Biochemical Reactions in *Neurospora*. *Proc. Natl. Acad. Sci. USA* **27**: 499-506.
10. Beadle, G. W. and E. L. Tatum. 1945. *Neurospora* II. Methods of Producing and Detecting Mutations Concerned with Nutritional Requirements. *Amer. J. Bot.* **32**: 678-686.
11. Muller, H. J. 1927. Artificial Transmutation of Genes. *Science* **46**: 84-87.
12. Stadler, L. J. 1928. Mutations Induced in Barley by X-Rays and Radium. *Science* **110**: 543-548.

13. Smith, H. O. and K. W. Wilcox. 1970. A Restriction Enzyme from *Hemophilis Influenzae*. I. Purification and General Properties. *J. Mol. Biol.* **51**: 379-391.
14. Linn, S. and W. Arber. 1968. Host Specificity of DNA Produced by *E. coli* X. *In vitro* Restriction of Phage ϕ d Replicative Form. *Proc. Natl. Acad. Sci. USA* **59**: 1300-1306.
15. Zoller, M. J. and M. Smith. 1982. Oligonucleotide-Directed Mutagenesis Using M13-Derived Vectors: An Efficient and General Procedure for the Production of point Mutations in any Fragment of DNA. *Nucl. Acids. Res.* **10**: 6489-6500.
16. Kunkel, T. A. 1985. Rapid and Efficient Site-Specific Mutagenesis Without Phenotypic Selection. *Proc. Natl. Acad. Sci. USA* **82**: 488-492.
17. Myers, R. M., L. S. Lerman and T. Maniatis. 1985. A General Method for Saturation Mutagenesis of Cloned DNA Fragments. *Science* **229**: 242-247.
18. Helinski, D. R. and C. Yanofsky. 1963. A Genetic and Biochemical Analysis of Second Site Reversion. *J. Mol. Biol.* **238**: 1043-1048.
19. Matteucci, M. D. and H. L. Heyneker. 1983. Targeted Random Mutagenesis: The Use of Ambiguously Synthesized Oligonucleotides to Mutagenize Sequences Immediately 5' of an ATG Initiation Codon. *Nucl. Acids Res.* **11**: 3113-3121.
20. Lim, W. A. and R. T. Sauer. 1989. Alternative Packing Arrangements in the Hydrophobic Core of λ repressor. *Nature* **339**: 31-36.
21. Lim, W. A. and R. T. Sauer. 1991. The Role of Internal Packing Interactions in Determining the Structure and Stability of a Protein. *J. Mol. Biol.* **219**: 359-376.
22. Min, K. T., M. H. Kim and D.-S. Lee. 1988. Search for the Optimal Sequence of the Ribosome Binding Site by Random Oligonucleotide-Directed Mutagenesis. *Nucl. Acids Res.* **16**: 5075-5089.
23. Hermes, J. D., S. C. Blacklow and J. R. Knowles. 1990. Searching Sequence Space by Definable Random Mutagenesis- Improving the

Catalytic Potency of an Enzyme. Proc. Natl. Acad. Sci. USA **87**: 696-700.

24. Wang, X. and G. J. Pielak. 1991. Temperature-Sensitive Variants of *Saccharomyces cerevisiae* Iso-1-cytochrome *c* Produced by Random Mutagenesis of Codons 43-54. J. Mol. Biol. **221**: 97-105.

25. Smith, G. P. 1985. Filamentous Fusion Phage: Novel Expression Vectors That Display Cloned Antigens on the Virion Surface. Science **228**: 1315-1317.

26. Hoogenboom, H. R., A. D. Griffiths, K. S. Johnson, D. J. Chiswell, P. Hudson and G. Winter. 1991. Multi-Subunit Proteins on the Surface of Filamentous Phage: Methodologies for displaying antibody (Fab) heavy and Light Chains. Nucl. Acid. Res. **19**: 4133-4137.

27. Cwirla, S. E., E. A. Peters, R. W. Barrett and W. J. Dower. 1990. Peptides on Phage: A Vast Library of Peptides for Identifying Ligands. Proc. Natl. Acad. Sci. USA **87**: 6378-6382.

28. Devlin, J. J., L. C. Panganiban and P. E. Devlin. 1990. Random Peptide Libraries: A Source of Specific Binding Molecules. Science **249**: 404-406.

29. Roberts, B. L., W. Markland, A. C. Ley, R. B. Kent, D. W. White, S. K. Guterman and R. C. Ladner. 1992. Directed Evolution of a Protein: Selection of Potent Neutrophil Elastase Inhibitors Displayed on M13 Fusion Phage. Proc. Natl. Acad. Sci. USA **89**: 2429-2433.

30. Arkin, A. P. and D. C. Youvan. 1992. Optimizing Nucleotide Mixtures to Encode Specific Subsets of Amino Acids for Semi-Random Mutagenesis. Bio/Technology **10**: 297-300.

31. Geysen, H. M., R. H. Meloen and S. J. Barteling. 1984. Use of Peptide Synthesis to Probe Viral Antigens for Epitopes to a Resolution of a Single Amino Acid. Proc. Natl. Acad. Sci. USA **81**: 3998-4002.

32. Fodor, S. P. A., J. L. Read, M. C. Pirrung, L. Stryer, A. T. Lu and D. Solas. 1991. Light-Directed, Spatially Addressable Parallel Chemical Synthesis. Science **251**: 767-773.

33. Lam, K. S., S. E. Salmon, E. M. Hersh, V. J. Hruby, W. M. Kazmierski and R. J. Knapp. 1991. A New Type of Synthetic Peptide Library for Identifying Ligand-Binding Activity. *Nature* **354**: 82-84.
34. Geysen, H. M., S. J. Rodda and T. J. Mason. 1986. *A Priori* Delineation of a Peptide Which Mimic a Discontinuous Antigenic Determinant. *Mol. Immunol.* **23**: 709-715.
35. Houghten, R. A. 1985. General Method for the Rapid Solid-Phase Synthesis of Large Numbers of Peptides: Specificity of Antigen-Antibody Interaction at the Level of Individual Amino Acids. *Proc. Natl. Acad. Sci. USA* **82**: 5131-5135.
36. Houghten, R. A., C. Pinilla, S. E. Blondelle, J. R. Appel, C. T. Dooley and J. H. Cuervo. 1991. Generation and Use of Synthetic Peptide Libraries for Basic Research and Drug Discovery. *Nature* **354**: 84-86.
37. Aarts, E. and J. Korst. 1989. Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing. John Wiley & Sons Ltd., Tiptree, Essex.
38. Goldberg, D. E. 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Publishing Company, Inc., New York.
39. Kirkpatrick, S., C. D. Gelatt and M. P. Vecchi. 1982. Optimization by Simulated Annealing. IBM. RC 9355.
40. Kirkpatrick, S., C. D. Gelatt and M. P. Vecchi. 1983. Optimization by Simulated Annealing. *Science* **220**: 671-680.
41. Cerny, V. 1985. Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm. *J. Opt. Theory Appl.* **45**: 41-51.
42. Eiben, A. E., E. H. L. Aarts and K. M. V. Hee. 1990. Global Convergence of Genetic Algorithms: A Markov Chain Analysis. *In* Parallel Problem Solving From Nature. H.-P. Schwefel and R. Manner. (Ed.) 4-12.
43. Holland, J. H. 1975. Adaptation in Natural and Artificial Systems. The University of Michigan Press, Ann Arbor.

44. Holland, J. H. 1973. Genetic Algorithms and the Optimal Allocation of Trials. *SIAM J. Comput.* 2: 88-105.
45. Holland, J. H. 1971. Processing and Processors for Schemata. *In* Associative Information Processing. E. L. Jacks. (Ed.) (American Elsevier, New York) 127-146.
46. Liepins, G. E. and M. D. Vose. 1991. Deceptiveness and Genetic Algorithms Dynamics. *In* Foundations of Genetic Algorithms. G. J. E. Rawlins. (Ed.) (Morgan Kaufmann Publishers, Inc., San Mateo, Ca.) 36-52.
47. Davidor, Y. 1991. Epistasis Variance: A Viewpoint on GA-Hardness. *In* Foundations of Genetic Algorithms. G. J. E. Rawlins. (Ed.) (Morgan Kaufmann Publishers, Inc., San Mateo, Ca.) 23-35.
48. Goldberg, D. E. and R. Lingle. 1985. Alleles, Loci, and the Travelling Salesman Problem. *In* Proceedings of an International Conference on Genetic Algorithms and Their Applications. J. J. Grefenstette. (Ed.) 154-159.
49. Davis, L. D. 1985. Applying Adaptive Algorithms to Epistatic Domains. *In* Proceedings IJCAI-85. (Ed.) 162-164.
50. Axelrod, R. 1987. The Evolution of Strategies in the Iterated Prisoner's Dilemma. *In* Genetic Algorithms and Simulated Annealing. L. David. (Ed.) (Pitman, London)

Chapter 2

The Evolution of a Digital Imaging Spectrometer

Digital Imaging Spectroscopy

The Digital Imaging Spectrometer (DIS) has undergone a major evolution in the last few years in its initial incarnation (pre-1987). It was meant to automatically identify fluorescent biological samples in 700-nanometer resolution. It was designed to be a portable, rugged, and easy-to-use instrument. The original design was based on a charge-coupled device (CCD) camera system. The camera system was used to capture the image of the sample. The image was then processed by a computer to identify the sample. The original design was based on a charge-coupled device (CCD) camera system. The camera system was used to capture the image of the sample. The image was then processed by a computer to identify the sample. The original design was based on a charge-coupled device (CCD) camera system. The camera system was used to capture the image of the sample. The image was then processed by a computer to identify the sample.

The original design was based on a charge-coupled device (CCD) camera system. The camera system was used to capture the image of the sample. The image was then processed by a computer to identify the sample. The original design was based on a charge-coupled device (CCD) camera system. The camera system was used to capture the image of the sample. The image was then processed by a computer to identify the sample. The original design was based on a charge-coupled device (CCD) camera system. The camera system was used to capture the image of the sample. The image was then processed by a computer to identify the sample.



Figure 2.1. Filter Array Spectrometer (after reference 1). This spectrometer utilized an complex image-processing algorithm.

2.1 The Evolution of a Digital Imaging Spectrometer

The Digital Imaging Spectrometer (DIS) has undergone a rapid evolution in the last six years. In its initial incarnation (pre-1987)¹ it was meant to automatically identify fluorescent bacterial colonies of *Rhodobacter capsulatus* (*Rb. capsulatus*) without the need for special infrared (IR) sensitive film. (*Rb. capsulatus* colonies fluoresce when the light-harvesting antennae, energy transfer proteins, are decoupled from the reaction center. (See below)) As such, it was a rapid functional assay for mutants made in the photosynthetic proteins. However, much of the functional and structural information of these proteins is encoded in the ground state absorption spectra of the intrinsic pigments (carotenoids, bacteriochlorophyll, and bacteriopheophytin) which are bound to these complexes.²

The next obvious step, then, was to convert the spectrometer into an absorption mode machine capable of obtaining spectra from colonies on a petri dish. The existing geometry of the light-source and camera made it reasonable to operate the machine as a reflectometer.

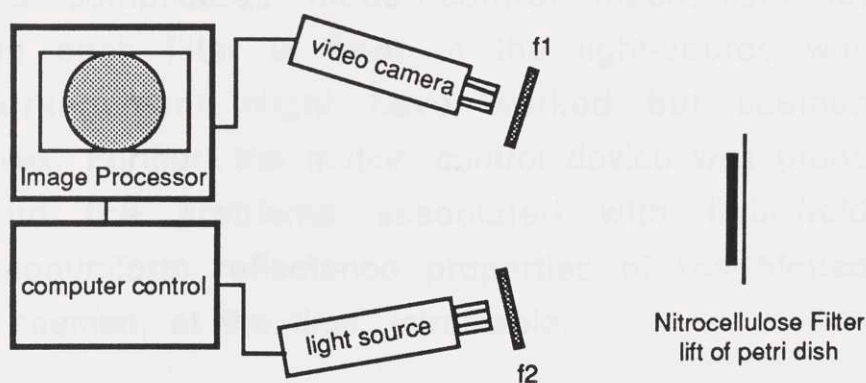


Figure 2.1. Filter Array Spectrometer (after reference ¹). This spectrometer utilized an complex image-processing chassis

controlled by a high-end Silicon Graphics computer. Different configurations of optical filters placed in front of the silicon-target video camera (f1) and/or the light source (f2) allow the spectrometer to obtain fluorescence emission, fluorescence excitation or ground-state reflectance spectra.

In this mode, a light-source (a 450W tungsten-halogen slide projector bulb) is filtered through fabry-perot interference filters and is used to illuminate a nitrocellulose lift of colonies from a petri dish. The lift technique was used to minimize glare and optical problems associated with colonies on agarose. A video camera, interfaced to a computer-controlled image processing chassis, was employed to image the nitrocellulose filter illuminated under different wavelengths and store the results on a hard disk. In the first report of the machine ¹, images obtained under three different wavelengths of light were used as the red, green and blue components of a computer-constructed, pseudo-color image. Those colonies which absorbed equally at all three wavelengths appeared as a shade of gray, and those which absorb any wavelength preferentially appeared colored. This technique composed a sophisticated version of the standard colorimetric assays commonly used in molecular biology such as the IPTG/XGal system.³

In order to extend the number of wavelengths available to the spectrometer an array of fabry-perot interference filters was constructed and a complicated motion-control mechanism for sequentially placing each filter in front of the light-source was contrived. This arrangement might have worked but seemed unnecessarily complex. Further, the motion control device was prone to malfunction and the problems associated with light-field homogeneity and nonuniform reflectance properties of the blotted colony morphology seemed, at the time, intractable.

The second paper written about this instrument⁴ reported a transmission mode device in which the interference filters dropped

into the light source like a standard projector slide. A pair of back-to-back opal glass diffusers were used to approximate a lambertian light field to illuminate a petri dish from behind.

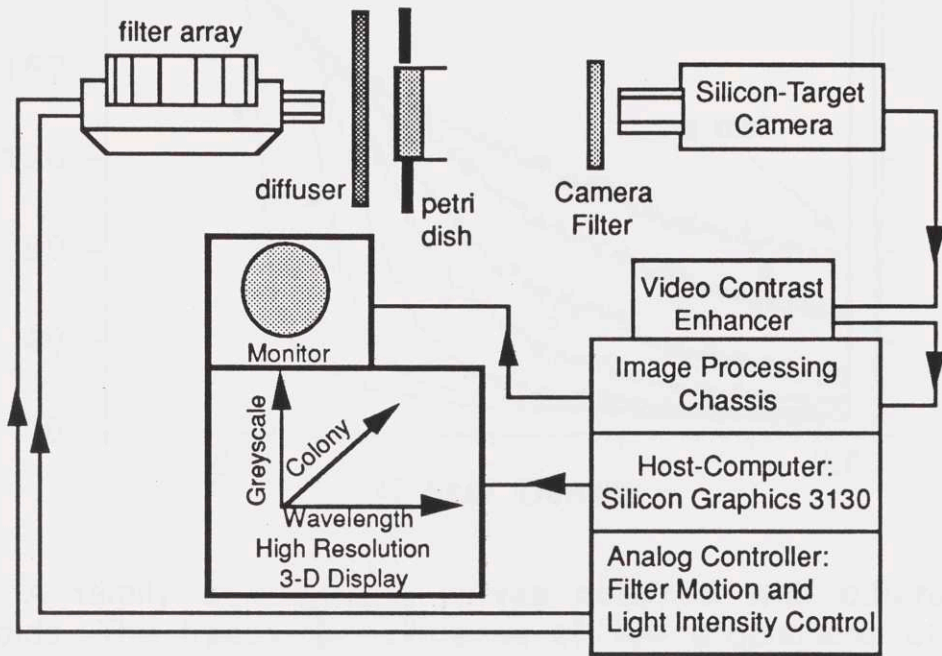


Figure 2.2. Hardware for the first generation Digital Imaging Spectrometer. The diffuser here is composed of a pair of back-to-back opal glass diffusers. Calibration for instrument response is achieved by setting light-source intensity such that a small "blank" region of the image always has a given grayscale. Calibration was performed by a computer-controlled, high-resolution dimmer attached to the light-source.

This turned out to be a poor approximation and the transmission of the pair was very low. Worse yet, the video camera was essentially capable of producing only eight bits of information per picture element (pixel). Aside from the fact that images had to be averaged because of noise in the camera, the silicon target tube and electronics had a highly non-linear response in the different optical density ranges.

Intensity Response Curves for DIS

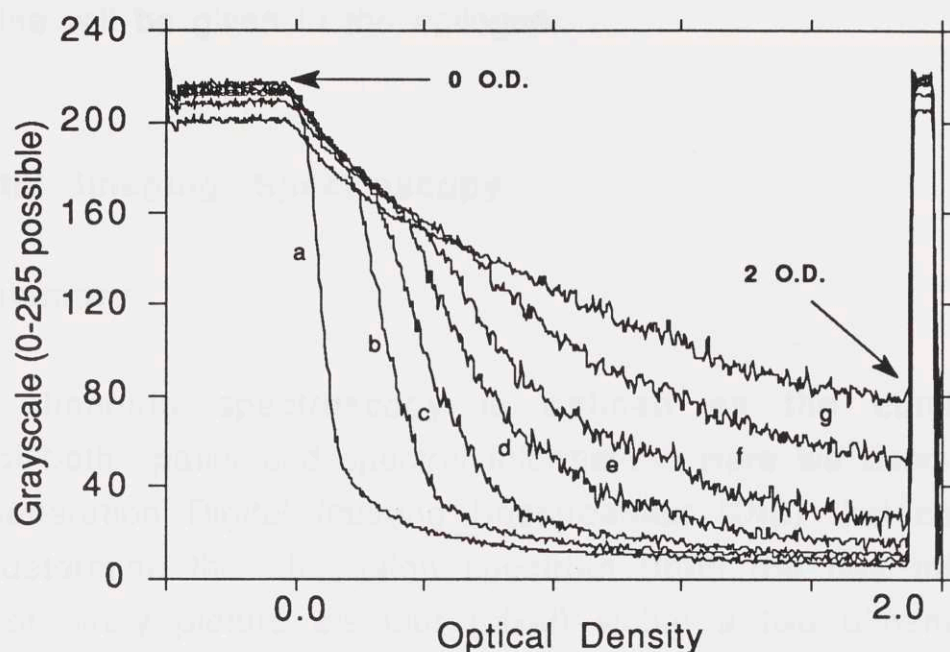


Figure 2.3. A family of response curves obtained with different video thresholds. The traces are averages of 100 acquisitions of a single pixel densitometric trace across the (0.0-2.0 OD) neutral density gradient. They demonstrate that the instrument response may be optimized to different optical density ranges. The setting of the video threshold has been decreased from curve (a) to curve (e). The flat portion at the far left and the spike at the far right of each response is a result of the densitometric trace extending beyond the ends of the gradient.

This made calibration very difficult and resulted in qualitative "grayscale" spectra instead of analytic absorption spectra. Nonetheless, the basic design logic shown in Figure 2.2 hasn't changed to the present date.

The following paper is a description of the current instrument, its technical specifications and its basic software. The paper is to appear in The Photosynthetic Reaction Center (J. Deisenhoffer, J.R. Norris, eds.) within the next year. A discussion of

the ongoing development and the current experimental application of the machine will be given in the epilogue.

2.2 Digital Imaging Spectroscopy

2.2.1 Summary

Imaging spectroscopy is defined as the combined analysis of both spatial and spectral information. Here we describe a second generation Digital Imaging Spectrometer (DIS) that can be used to determine the absorption spectrum (from the near-infrared to blue) of every picture element (pixel) within a two dimensional target. Using a 384 x 576 charge-coupled device as a detector, this instrument is operationally equivalent to 200,000 conventional spectrometers running in parallel. Image processing techniques are used to extract features for spectral determination and comparison. For example, from a single petri dish, over 1000 colony spectra can be acquired at 5 nm resolution from 400 to 930 nm. Relatively noiseless spectra have been obtained on colonies expressing very low levels of pigments (<0.01 OD peak absorbance). A Light Harvesting I antenna mutant from *Rhodobacter capsulatus*, which undergoes spectral "splintering", is used to demonstrate the utility of this new instrument in the field of molecular genetics.

2.2.2 Machine Vision for Library Screening

In response to an inability to predict the effects of a given mutation, random mutagenesis has been gaining prominence as a method to explore thousands of mutants in parallel.⁵⁻⁹ Artificial intelligence (AI) methods have been proposed to guide the construction of quasi-random mutant libraries.^{10, 11} (see Chapters

3,4,5 and 6) In order to assay complex libraries of mutants, rapid screening techniques must be developed. Screening petri dishes by optical spectroscopy^{1, 4} is perhaps the most rapid and informative method for assaying the phenotypes of mutants that express chromogenic substances (e.g. pigmented proteins) or react with extrinsic indicator dyes. AI based mutagenesis and machine vision techniques are currently being applied to the reaction center (RC) and light harvesting (LH) antennae of photosynthetic bacteria.¹²⁻¹⁴

Since the ground state absorption spectra of the RC and LH antennae are excellent indicators of expression and functionality of these pigment-protein complexes, we have developed an instrument capable of obtaining spectra from all of the colonies on a petri dish in less than 20 minutes. Prior to the invention of the Digital Imaging Spectrometer (DIS), the protein from each colony would have to be isolated separately in order to obtain its spectrum. This conventional process takes months of microbiological and biochemical preparation time, in addition to the time required to record and compare thousands of individual spectra.

Digital imaging spectroscopy relies on image processing¹⁵ and radiometric calibration techniques to reconstruct spectra from features within a two-dimensional target imaged at a number of wavelengths. This instrument has evolved considerably since it was first described.^{1, 4} This chapter includes a detailed description of the hardware and software used in the 2nd generation device. To date, DIS technology has been used to determine the spectra of a variety of biological targets, including highly pigmented colonies from photosynthetic bacteria. Here, we also describe the construction of an ideal "tester strain" for DIS analysis that rapidly mutates to produce a variety of spectrally distinct phenotypes.

2.2.3 Construction of a Spectral "Tester" Strain

The central (LHI) antenna of *Rhodobacter capsulatus* is composed of two transmembrane α -helices labelled α and β which are 58 and 49 amino acid residues in length, respectively.¹⁶ The hydrophobic regions of these helices are approximately 20 residues long, sufficient for one transmembrane domain. An α - β peptide dimer binds a pair of electronically coupled bacteriochlorophylls (BCHs) and at least one carotenoid.¹⁷ A conventional ground state absorbance spectrum of LHI can be found in reference 18. The BCH pair absorbs at 870nm and the carotenoids absorb between 400 and 500nm. The wild-type sequences for the α and β subunits of LHI have been tabulated for a variety of species.¹⁹ The BCHs are thought to be bound at Alanine-X-X-X-Histidine motifs (where the X's represent hydrophobic amino acid residues) within the transmembrane regions of the α and β subunits.

In the vicinity of the BCH binding site, five amino acid residues were changed to leucine, resulting in a run of 14 residues of which only two are not leucine (Figure 2.4). The putative axial histidine ligand and the -4 alanine residue were conserved to preserve the BCH binding site motif. These mutations were considered conservative since leucine appears frequently in the phylogeny.¹⁹

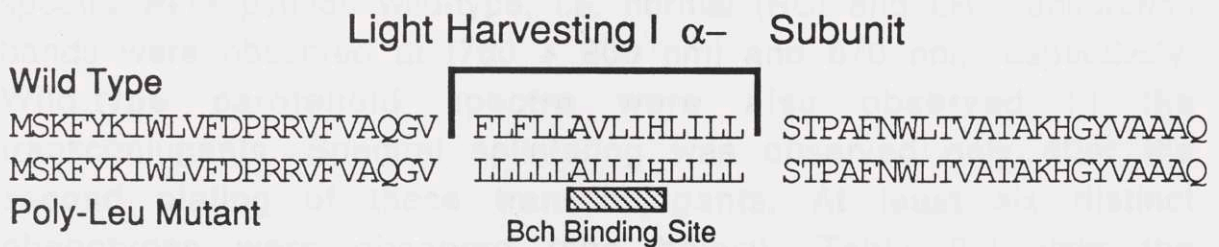


Figure 2.4. LHI Wild-Type and Mutant Sequences. The single

letter code for the amino acid residues is used to indicate the sequence of the LHI a subunit from *Rb. capsulatus* and a mutant derivative. Five simultaneous mutations in the "poly-Leu" mutant result in 12 out of 14 leucine residues in the vicinity of the putative binding site for the bacteriochlorophyll dimer (B870). The poly-Leu mutant is unstable and "splinters" into a variety of colorful segregants that can be used as a test target for the digital imaging spectrometer.

All standard cloning procedures were essentially as described by Sambrook *et al.*²⁰ M13 phage were maintained in *E. coli* strain MV1190 and all plasmid pU2925^{21, 22} derivatives were maintained in the conjugal *E. coli* strain S17-1. Plasmid pU2925 mutant was conjugated from S17-1 donors. Mutagenesis was performed on M13(A α 28E), an M13mp18 derivative containing the 490 base-pair *Hind* III - *Eco*R I fragment from pU2925. Success of the mutagenesis was confirmed by the loss of the *Xho* I site (present in the DNA sequence encoding the A α 28E mutation) and by DNA sequencing. After cloning the fragment from M13 to pU2925, the plasmid was electroporated into S17-1. The transformed plasmid was verified by the absence of the marker *Xho* I site. The plasmid containing the LHI leucine mutations will hereafter be referred to as pU2925(α -poly-Leu) or the "poly-Leu" plasmid.

Following conjugation of the poly-Leu plasmid into *Rb. capsulatus*, strain U43^{21, 22}, the transconjugants were selected on RCV/kan media under dark conditions at 31°C. Spectra from approximately 500 colonies were obtained using the DIS. All colony spectra were pseudo wild-type, i.e. normal [RC] and LH I absorption bands were observed at [760 + 800 nm] and 870 nm, respectively. Wild-type carotenoid spectra were also observed in the transconjugants. Spectral splintering was observed only after the second plating of these transconjugants. At least six distinct phenotypes were observed (see below). Table 2.1 lists the phenotypes for these mutants. The peaks observed at 630nm and

670nm were tentatively identified as two different BCH precursors.²³ Plasmids from each of these phenotypic categories of segregants were isolated and re-conjugated into virgin U43 to identify the locus of diversity. None of the segregant (i.e. spectrally splintered) phenotypes were carried on the isolated poly-Leu plasmid. This was verified by sequencing the *Hind* III - *Eco*R I region of each mutant type and showing that only the engineered poly-Leu mutations were present.

Mutant	crt	LHI	RC	bch(630)	bch(670)
Poly-Leu	+	+	+	-	-
Segregant-1	+	-	-	-	-
Segregant-2	+	-	+	-	-
Segregant-3	-	+	+	-	-
Segregant-4	+	+	+	-	-
Segregant-5	-	-	-	+	-
Segregant-6	-	-	-	-	+

Table 2.1. Phenotypes of Spectrally Splintered Mutants. At least six phenotypically distinct segregants of the poly-Leu mutant have been identified after replating this unstable strain. A variety of bacteriochlorophyll or carotenoid biosynthesis mutations occur in the segregant backgrounds which apparently stop the coexpression of the poly-Leu structural motif in the presence of carotenoids. The exception to this observation is segregant #4 which appears to be a stabilized form of the poly-Leu mutant with an unmapped pleiotropic mutation in the chromosomal background.

Colonies resulting from the conjugation of the poly-Leu plasmid into a carotenoidless background (U43b)²⁴ did not segregate into the spectrally splintered phenotypes upon replating. These transconjugants expressed both LHI and RCs in the absence of any carotenoid. Furthermore, photosynthetic growth of U43(pU2925(α -poly-Leu)) did not exhibit splintering and showed a crt⁺, RC⁺, LHI⁺

phenotype. When these bacteria were resuspended, replated, and grown under aerobic, dark conditions, no splintering occurred; however, expression of the proteins was low. The splintering behavior, therefore, is thought to be due to a severe growth disadvantage (under permissive conditions) for strains expressing both the poly-Leu mutation and carotenoids. Upon repurification, the background mutates to block carotenoid expression, or LHI expression is lost due to *bch* mutations. However, one category of segregants (i.e. "segregant-4", see Table 2.1) appears to have stabilized and is pseudo wild-type in appearance.

The splintering of the poly-Leu mutant into spectroscopically distinct phenotypes makes it an ideal tester strain for demonstrating DIS analysis. The remainder of this chapter is concerned with the technical characteristics of the DIS.

2.2.4 DIS Hardware Specifications

A schematic representation of the DIS is shown in Figure 2.5.

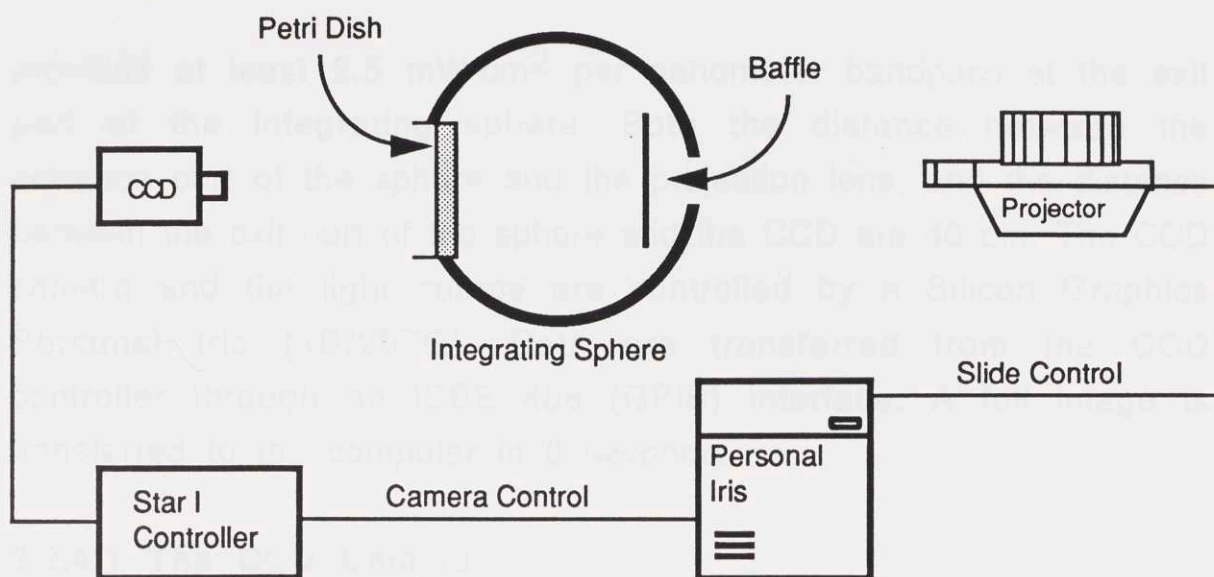


Figure 2.5. DIS Schematic. The digital imaging spectrometer uses a variety of off-the-shelf hardware components in conjunction with a complex computer program. Image processing and spectroscopy are combined such that the spectrum of any feature (e.g. bacterial colony) can be obtained from the near-infrared to the blue at 5 nm resolution. Very smooth spectra can be obtained on chromogenic substances with as little as 0.01 OD peak absorbance.

In this geometry, a Peltier cooled Photometrics Star I CCD camera (384x576 pixels) is used to image a petri dish mounted in the exit port of an 12 inch diameter integrating sphere. The sphere is used as a high throughput, flat diffuser to provide a uniform light field across the petri dish. The sphere transmits approximately 30% of the light entering its rear port over the entire spectral range. The light source is a modified Kodak slide projector which contains a set of Fabry-Perot interference filters mounted in blank slide holders. Custom filters were constructed with either 5 nm or 10 nm bandpass (full width at half the transmission maximum (FWHM)) between 930 and 400 nm. Filters were selected with greater than 50% transmission at the center wavelength and less than ± 1 nm error in the wavelength of maximum transmission. The light source

provides at least 2.5 mW/cm^2 per nanometer bandpass at the exit port of the integrating sphere. Both the distance between the entrance port of the sphere and the projection lens, and the distance between the exit port of the sphere and the CCD are 40 cm. The CCD camera and the light source are controlled by a Silicon Graphics Personal Iris (4D/25TG). Data are transferred from the CCD controller through an IEEE 488 (GPIB) interface. A full image is transferred to the computer in 6 seconds.

2.2.4.1 The CCD Camera

The 1st generation instrument reported in reference 4 employed a Dage model 68 silicon target video camera. Advantages of a cooled, low-light CCD camera over the silicon target camera are numerous:

(1) The major advantage is that the CCD camera is a photon counter that has a linear response.²⁵ After an a image is subtracted (representative of dark current and variation in the pixel mosaic), the number of electrons read are directly proportional to photons reaching the detector (within noise limits; see below). The proportionality constant is the wavelength dependent quantum yield of the CCD (data available from manufacturer). This yield is rather good over the visible and near-infrared (VIS/NIR) range, but extended ultra-violet sensitivity requires the deposition of a special coating on the CCD. While the response of the DIS to wavelength can be normalized, it is essential that the CCD provide a linear correlation between photons counted and light dose (intensity \times time) for a given wavelength. In our application, ratioing the photons counted for pixels within a feature (such as a colony) to the photons counted from a "blank" area is analogous to to the calculation of I/I_0 in a double-beam absorption spectrometer.

Previously, when using silicon-target cameras, it was necessary to repeatedly adjust the light source's intensity until the selected "blank" pixels matched a pre-determined grayscale value. This required up to 20 time consuming adjustments per filter in the first generation instrument. This was necessary because the response of the silicon-target camera was highly non-linear, making it difficult to accurately calculate the light intensity necessary to achieve the target grayscale.

(2) The dynamic range of the CCD camera is 12 bits (0-4095 grayscale), increasing the range of the camera 16 times over the silicon target camera, which utilized an 8-bit (0-255 grayscale) analog-to-digital frame grabber. Other CCD cameras are currently available with 16 bits of dynamic range.

(3) Using a video camera, noise is apparent even on an 8 bit grayscale. With the exposure levels and integration times we currently use, the expanded 12-bit grayscale of the cooled CCD is essentially noiseless. There are four basic sources of noise in the CCD system: a) Photon Shot Noise (PSN) is a direct result of the quantum nature of light. It is well-known that the total number of photons emitted by a steady-source fluctuates according to the Poisson distribution. The number of photoelectrons collected by a CCD follows the same statistics. For relatively short integration times (and/or low-light levels), the PSN is lower than preamplifier noise and the dark-current error. For a given CCD (and thermal conditions), a crossover point occurs where PSN becomes the limiting factor at higher light doses. In our DIS application, we are in this latter domain, wherein PSN dominates. The Star I CCD²⁶ stores up to 160,000 electrons/pixel, yielding 39 electrons per grayscale unit. Approximately 117,000 photoelectrons per pixel (grayscale ca. 3000), yields about 342 electrons of noise per pixel

by the $N^{1/2}$ law. b) Preamplifier noise can be reduced by cooling the on-chip preamplifier to $-45\text{ }^{\circ}\text{C}$. This type of noise can be reduced to a few electrons/pixel/sec. It is therefore much less than the PSN and not significant in our application. c) Read-out noise of the D/A converter is dependent on the readout rate. In our system, this noise is approximately 25 electrons RMS/second; again, less than the PSN. d) Dark-current with a Star I CCD cooled to $-45\text{ }^{\circ}\text{C}$ is approximately 15 electrons/pixel/second. Integration times for our application rarely exceed 15 seconds. Under these conditions, the error associated with the dark current is $(15 \times 15)^{0.5} = 15$ electrons/pixel. Accounting for all sources, we estimate the total noise per pixel in our system at 400 electrons, which converts to a grayscale error of approximately ± 5 . Noise is further reduced by the square-root of the number of pixels averaged. Thus, if the grayscale values of 100 pixels within a certain feature are averaged, then the noise is reduced by a factor of 10. This converts to an error of about 1 part per 4096.

(4) The 4096 levels of grayscale can be used to obtain a differential sensitivity of less than 0.001 OD up to 1 OD total absorbance. Using the Beer-Lambert Law: $10^{-A} = I/I_0 = 4095/4096 = 0.0001$, while at 1 OD, $I/I_0 = 408/409 = 0.001$ OD. Noise inherent in the average of 100 pixels (ca. 1 grayscale unit) is calculated in a similar manner and yields 0.0004 OD noise at 0 OD and 0.004 OD at 1 OD. Experimentally, we have observed noise levels that are consistent with these calculations. Very smooth spectra have been obtained from colonies expressing LH antennae¹² and RCs¹³ with peak absorbances of 0.01 OD.

(5) Variable integration times are a feature of the CCD camera that can be used in conjunction with predetermined light-levels (as a function of wavelength) to partially equalize the

response of the system. System response is determined by the transmission profile of the filtered light source and the CCD. We use a standard tungsten source whose spectrum is that of a black-body. Light intensity for such a source is much lower in the blue than the red, therefore integration times are commensurately longer for blue wavelengths.

2.2.4.2 The Light Source and Integrating Sphere

Monochromatic light is generated in the DIS by using a series of Fabry-Perot interference filters in a modified slide projector (102mm f/2.8 projection lens). This source is essentially as described in reference 4. However, the voltage to the WKO tungsten-halogen lamp is now set to a constant 62 volts. The projector's heat filter has been removed to increase throughput in the NIR. It was therefore necessary to protect the filter holders with an aluminum backing to minimize the effects of the increased heat. The filters are Fabry-Perot interference filters spaced at either 10 nm increments with a 10 nm bandpass or at 5 nm increments with a 5 nm bandpass (FWHM). Wavelengths between 930 nm and 400 nm are routinely scanned. Light passing through the filters is focused onto the entrance port of the integrating sphere. The power of the light source at the entrance port is about 400 mW/cm² (measured at 640 nm for a 10 nm bandpass filter with a Li-COR, inc. PAR meter, model LI-185B using a "QUANTUM" detector).

Since the absorption spectra of colonies are calculated by the ratio of the intensity of light within a colony to that of a reference area somewhere on the petri dish, the uniformity of the light field is an important variable. A custom designed 12" diameter integrating sphere serves as a uniform diffuser for the light source. The inside of the sphere is coated such that between 400 nm and

1100 nm, the surface has a reflectance value above 97%. The sphere contains two diametrically placed ports; a 3.81 cm diameter entrance port and an 8.26 cm exit port in which a petri dish may be seated. A circular baffle is placed between the entrance and exit ports to block direct illumination of the exit port by the light source. The baffle sits approximately 10 cm away from the entrance port. Without the baffle in place and assuming perfectly uniform reflectance within the sphere, the theoretical throughput²⁷ of the sphere may be calculated by:

$$\tau = \frac{\rho f_e}{1 - \rho(1 - f_j)} \quad \text{Eq. 2.1}$$

where τ is the throughput, ρ is the sphere reflectance, f_e is the area of the exit port divided by the surface area of the sphere and f_j is the sum of all the port areas divided by the surface area of the sphere. For $\rho = 0.975$, equation 1 predicts a 55% throughput for the sphere geometry described above (minus the baffle). For the 640 nm filter approximately 4.5 mW enters the sphere. At the exit port, the light intensity was measured at 25 mW/cm² (i.e. 1.4 mW exited the sphere). This yields a throughput of 31%. The 24% deficit in throughput is most probably due to the placement of the baffle in the integrating sphere. When light enters the sphere, it immediately hits this Lambertian surface and some light is reflected back out the entrance port. In addition, the baffle adds to the internal surface area of the integrating sphere.

2.2.5 DIS Software Specifications

The control and analysis software used to obtain a spectrum is written in the C language. This program can be divided into three major parts: 1) Image Analysis (IA), 2) Spectral Acquisition (SA), and 3) Classification and Display (CaD). A flow chart of the experimental procedure is shown in Figure 2.6.

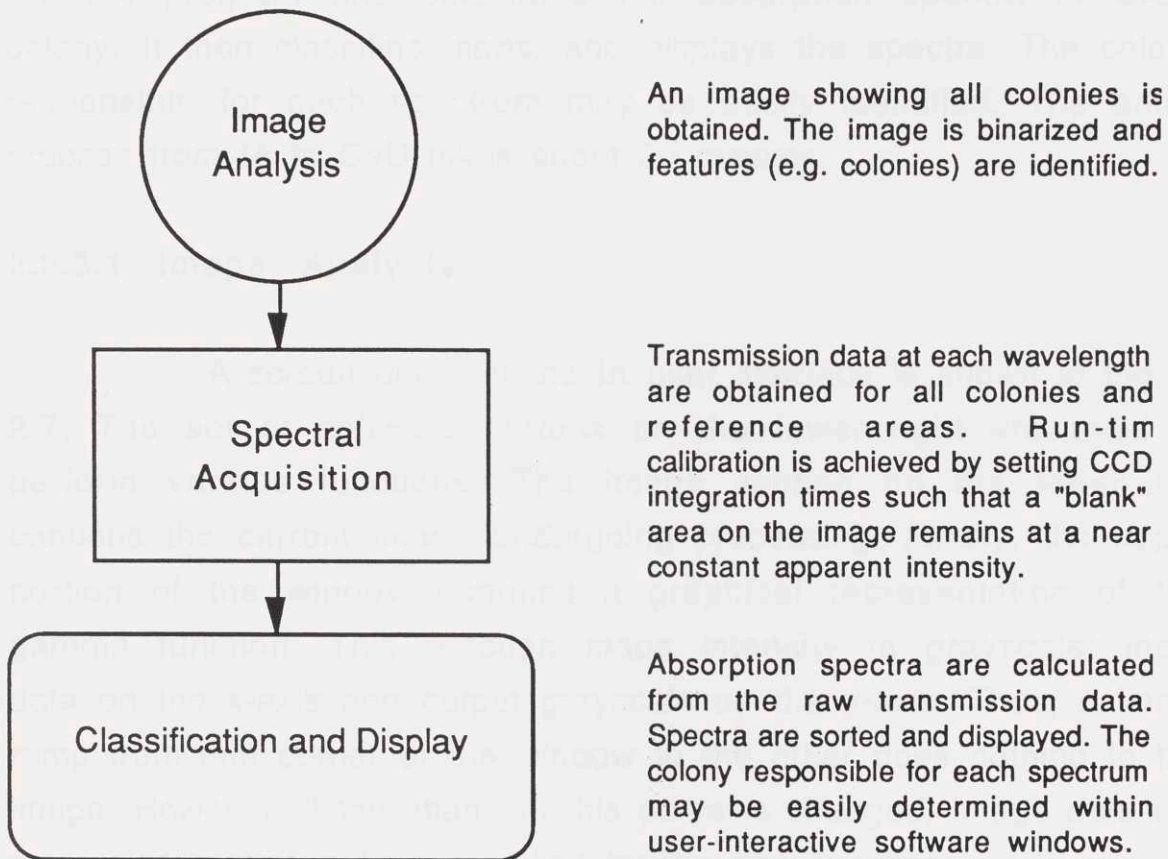


Figure 2.6. DIS Experimental Flow-Chart. Hardware and software functions act in conjunction to yield processed spectra from a series of monochromatic images. This process can be divided into three logical stages as shown in the flowchart. Each of these stages is described in more detail in the text and the figures which follow.

Initially, a petri dish is placed in the exit port of the integrating sphere. Using IA software, an image of the plate is taken under illumination with a filter at which all the colonies absorb or

scatter. The IA software identifies the center, average radius, and perimeter of each colony. The SA program then uses this information to obtain data only from these areas of the image. Data are collected for each colony at every wavelength within the filter set. Reference and instrument response corrections are performed during run-time as described below. The CaD program processes the raw data from the SA program and calculates the absorption spectra of every colony. It then classifies, sorts, and displays the spectra. The colony responsible for each spectrum may be easily identified. The entire process from IA to CaD takes about 20 minutes.

2.2.5.1 Image Analysis

A screen dump of the IA user interface is shown in Figure 2.7. The set of software buttons on the lower right are used to perform various functions. The image section on the lower left contains the current image undergoing processing. Finally, the upper portion of the window contains a graphical representation of the gamma function. This function maps intensity to grayscale: input data on the x-axis and output grayscale on the y-axis. Thus, a linear ramp from one corner of the window to the other does nothing to the image. However, if the shape of this curve is changed, image data are remapped to new values specified by the new function.

A flow chart of the IA procedure is shown in Figure 2.8.

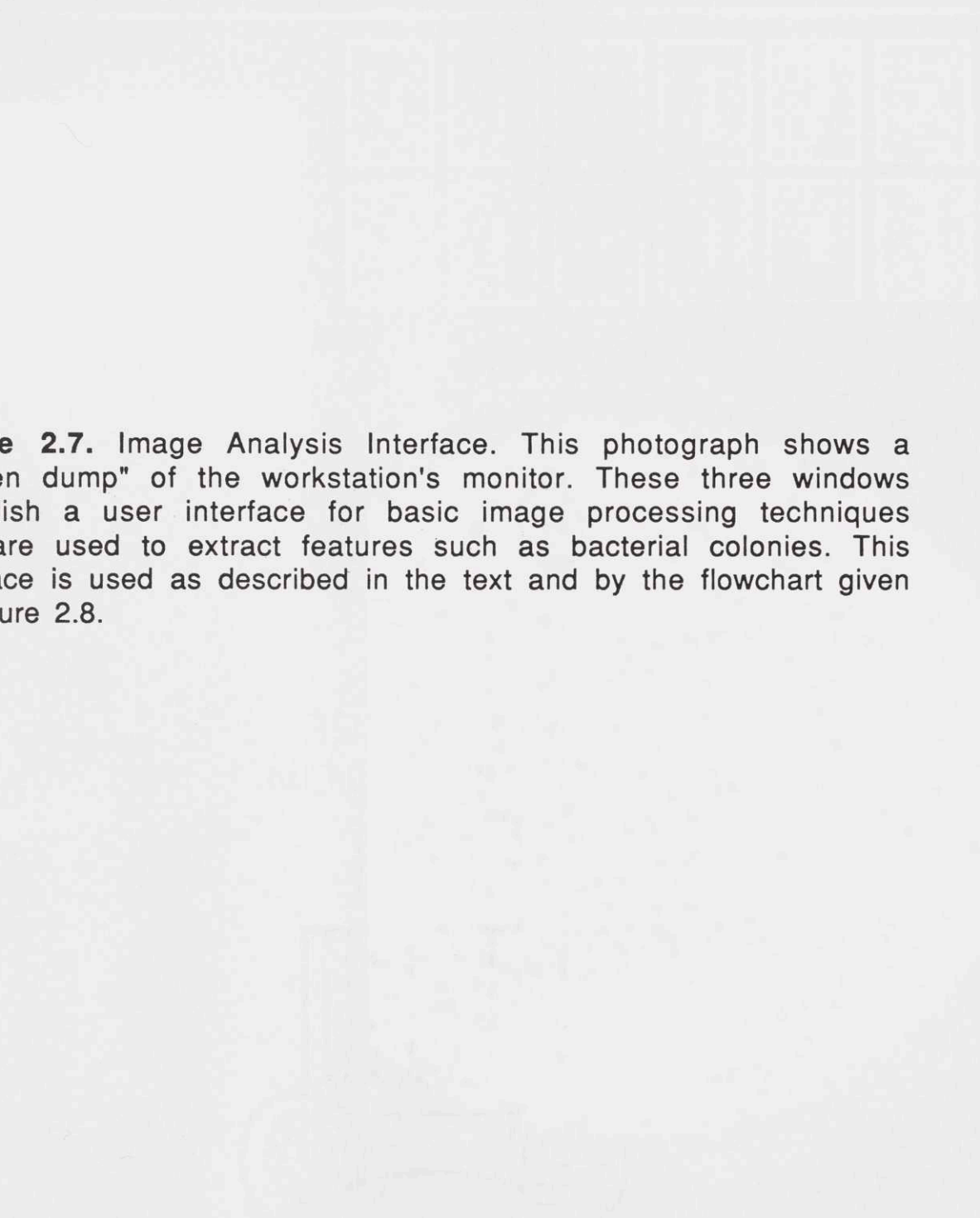
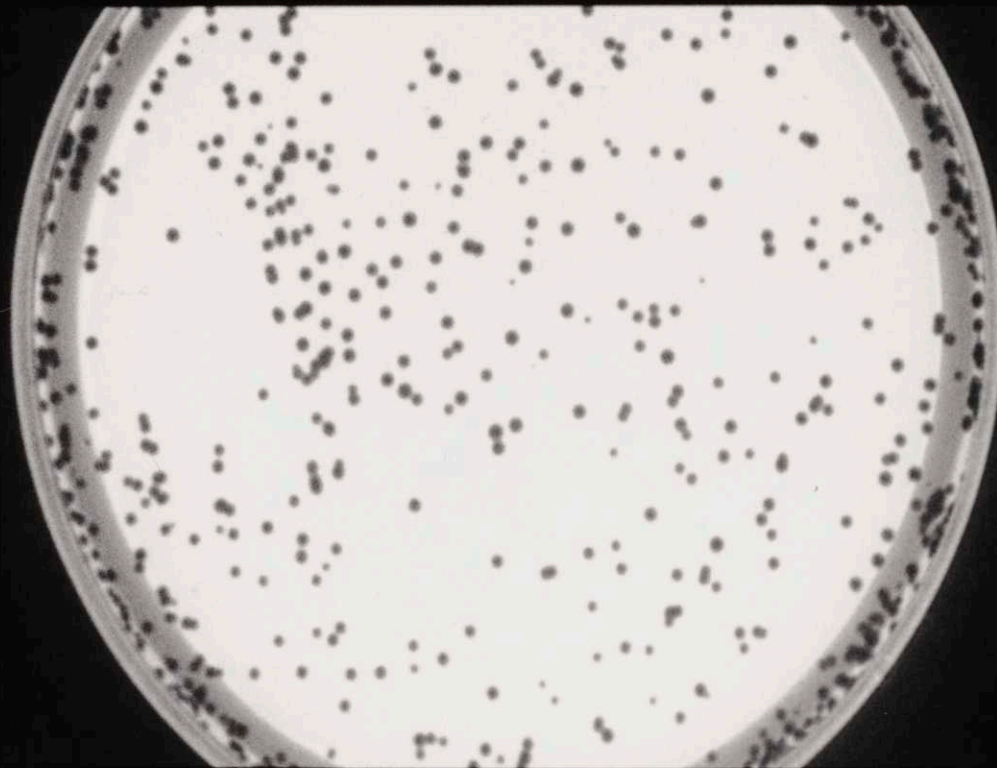
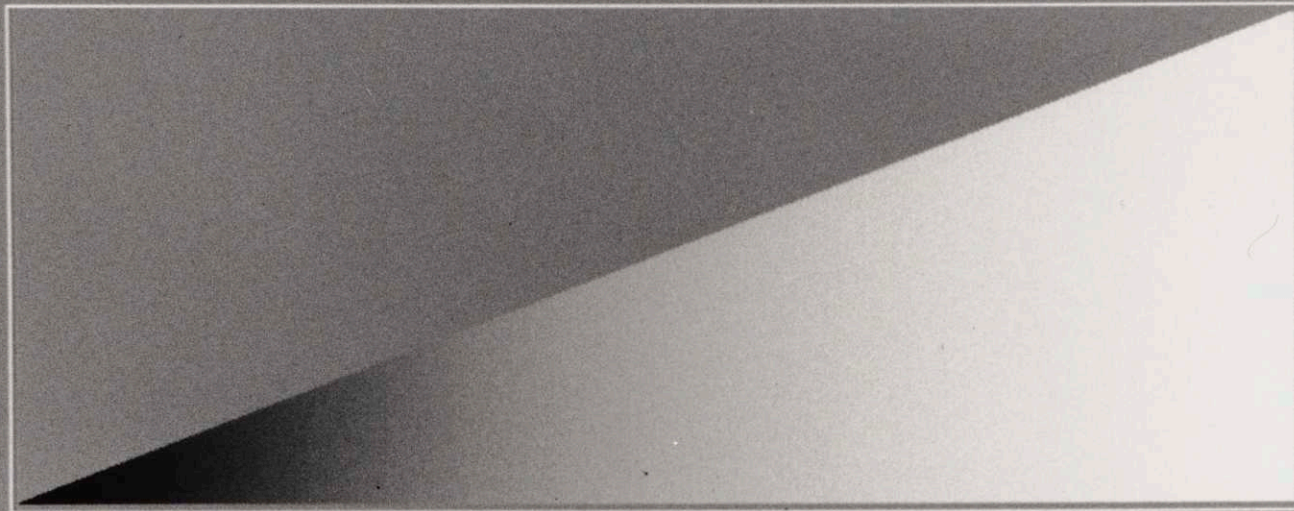


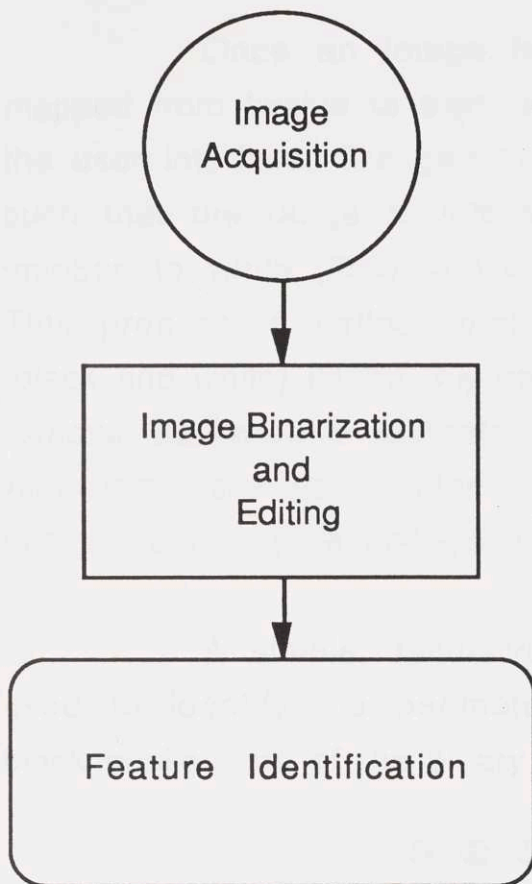
Figure 2.7. Image Analysis Interface. This photograph shows a "screen dump" of the workstation's monitor. These three windows establish a user interface for basic image processing techniques that are used to extract features such as bacterial colonies. This interface is used as described in the text and by the flowchart given in Figure 2.8.



Load	Save
Stretch	Seed Fill
Line+	Line-
Overlays	Reference
Undo	Take Image
Apply Gamma	Quit



28



An image of the petri dish is obtained. The dish should be illuminated by a wavelength of light at which all the colonies show significant absorbance. Blue wavelengths are generally used since all the colonies will exhibit relatively high scattering.

A "gamma function" is used to map all the grayscale values which make up the images of the colonies to white and all other grayscale values to black. The resulting binary (black and white) image is then edited to eliminate background artifacts and to separate overlapping colony features.

A "feature extraction" algorithm is used which identifies the perimeter of each colony. The colony center and average radius are calculated. This information is used by the spectral acquisition and CaD programs to calculate spectra.

Figure 2.8. Image Analysis Flow-Chart. Used in conjunction with the user interface displayed in Figure 2.7, this flow-chart establishes the procedure by which features are extracted from raw image data. The logic for the feature extraction algorithm used in DIS (i.e. the lower box in this flow-chart) is expanded upon in the text and Figures 2.10 and 2.11.

The goal of IA is to identify the location of colonies on the petri dish. This is achieved in three steps: 1) image acquisition, 2) image binarization and editing, and 3) feature extraction. The image to be processed should be acquired from a petri dish illuminated under a wavelength of light at which all the colonies have high contrast against the background. Generally, this occurs at shorter wavelengths where the scattering component is largest. In this case, the features will appear dark against a white background.

Once an image has been obtained it is automatically mapped from twelve to eight bits for display in the image section of the user interface. The gamma function is then modified by the user such that the range of intensities contained within the colonies is mapped to white (255) and everything else is mapped to black (0). This process is called "binarization", since it produces a binary (black and white) image (Figure 2.9). The image can then be edited to remove background artifacts and to separate connected colonies. Algorithms are now under development which will fully automate this procedure. (See epilogue.)

A simple, recursive feature extraction algorithm is then used to identify the perimeter of the white colonies against the black background of the binary image.

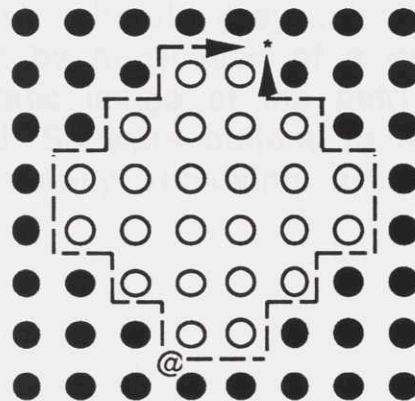


Figure 2.10. Pixel Level Image of a "Colony" for Feature Extraction. After binarization, a colony appears to be a group of white pixels on a black background. Tracing the edge of the colony and generating a list of the coordinates of these perimeter pixels constitutes a feature extraction. The algorithm for feature extraction is described in the text and summarized by the flow-chart in Figure 2.11. The first edge pixel found in the trace is designated by an "@"; the perimeter of the feature is closed by the algorithm at pixel "*".


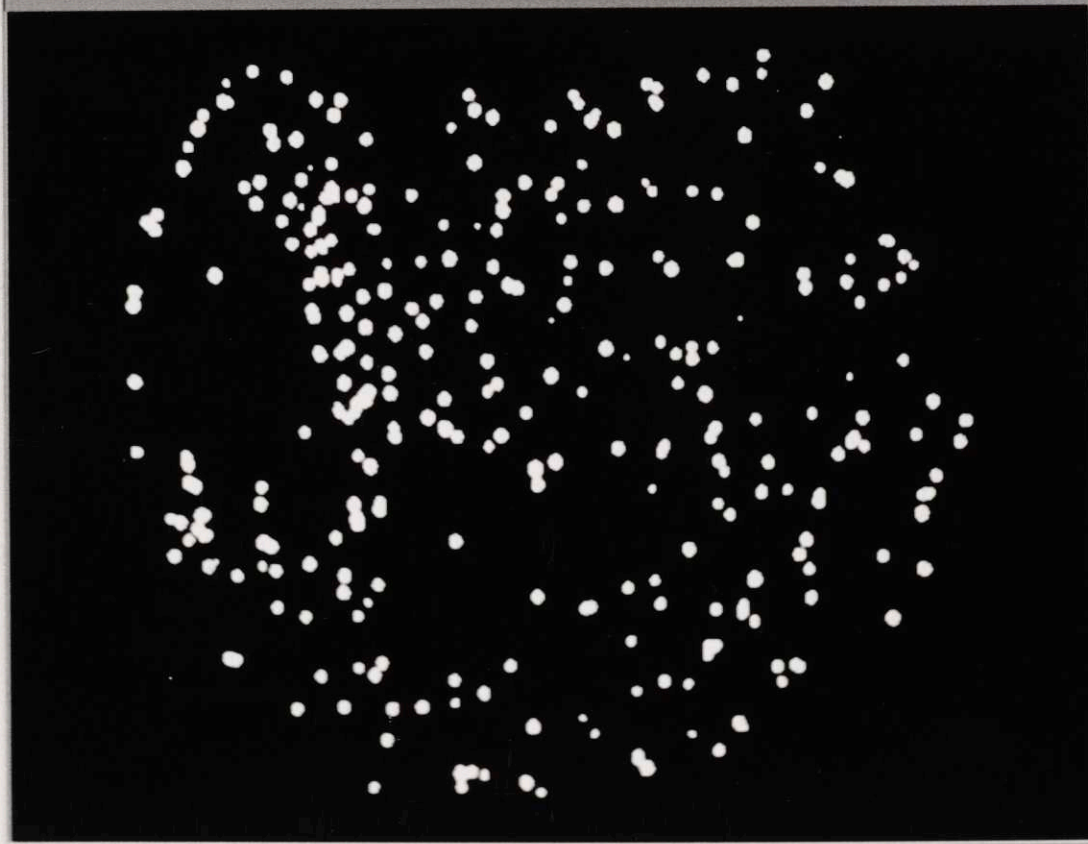


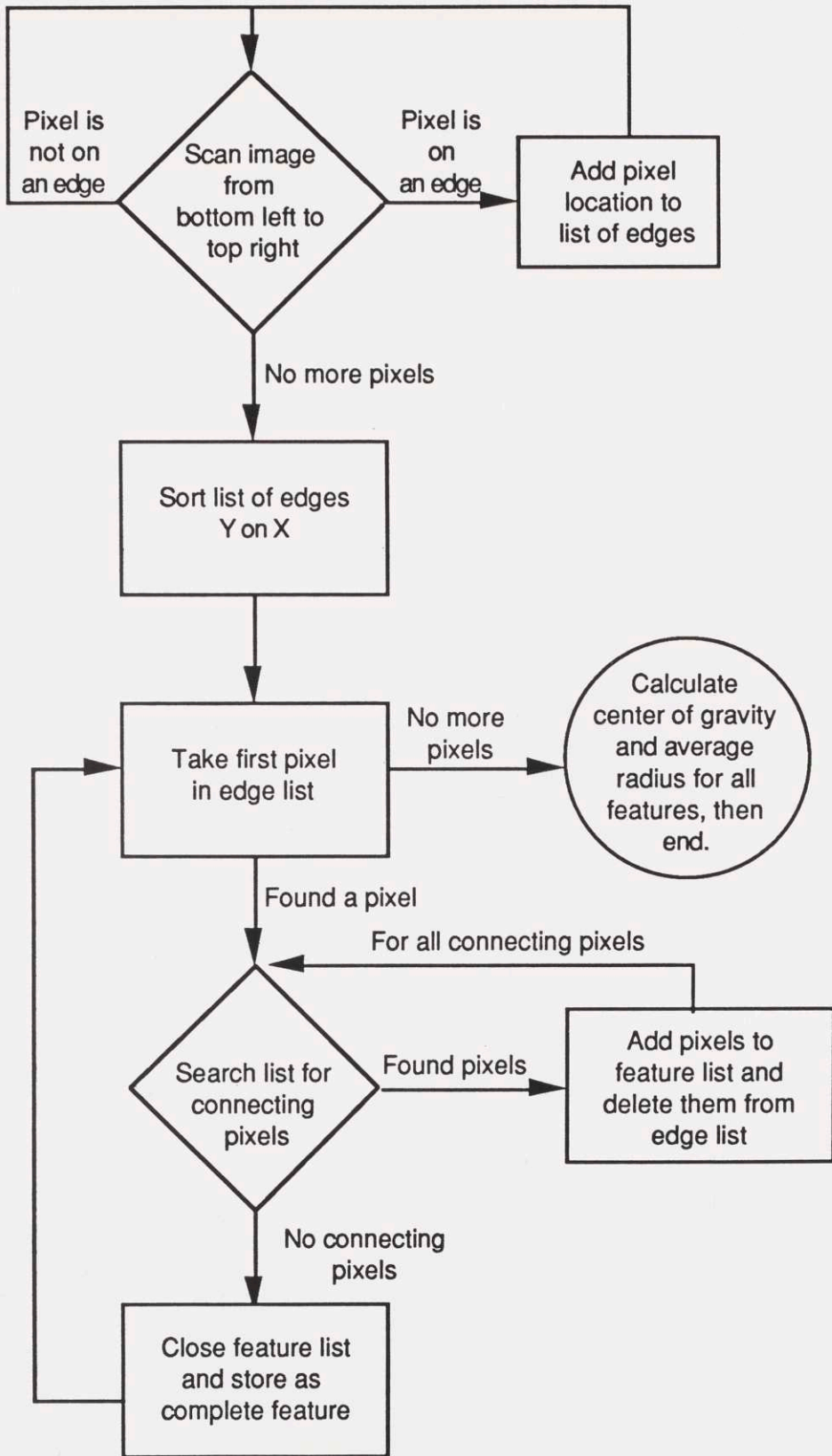
Figure 2.9. Binary Image. Feature extraction algorithms typically process "binary images", wherein grayscale images are mapped to black-and-white images by application of a gamma function. In this figure, the monochromatic image of the petri dish shown in Figure 2.7 has been binarized. Software buttons to the right also provide a user interface for manually removing artifacts and for isolating overlapping colonies.



Load	Save
Stretch	Seed Fill
Line+	Line-
Overlays	Reference
Undo	Take Image
Apply Gamma	Quit
<input type="checkbox"/>	<input type="checkbox"/>

42

Figure 2.11. Feature Extraction Flow-Chart. The feature extraction routine outlined in this flow-chart is specially suited for tracing "blobs" such as colonies. The edge pixels around a colony, and all of the pixels within this perimeter constitute a "feature". DIS utilizes the spatial coordinates of each feature to process data from pixels within a single object (e.g. a colony).



A flow-chart for the algorithm is shown in Figure 2.11. The extractor begins by searching for an edge point, i.e. a white point who has at least one (of a possible eight) black neighbor. If it hasn't visited this pixel before it looks for all neighboring edge points and then all their neighboring edge points and so on, until it can find no more edge neighbors. The set of connected edge points starting with the first is called a feature. This procedure continues until all edge points have been located. The center-of-mass, average radius, and perimeter of each feature is stored on disk and passed to the SA program. In some cases, where a substance is excreted by the colonies, it is a ring around the feature which is of interest. A reference area on the image (i.e. a region of clear agar) is also chosen and passed to the SA Program.

2.2.5.2 Spectral Acquisition

The spectral acquisition display is shown in Figure 2.12. The three sections are: 1) wavelength vs. reference intensity region in the lower left, 2) wavelength vs. integration time region in the upper left, and 3) the raw spectrum region at the right. This program requires a number of simple parameters. First, the target reference grayscale intensity is set. The intensity of the reference area must achieve this value, within specified limits (the target range), before data from the colonies are downloaded. The higher this blank value is set, the larger the possible dynamic range of the spectra; however, acquisition time will be proportionately longer. At this point it is decided by the operator whether to use a single reference or a small area around each feature as the reference for each object. Using local references corrects for global inhomogeneities in the light field. The number of pixels to be averaged in each object is also specified.

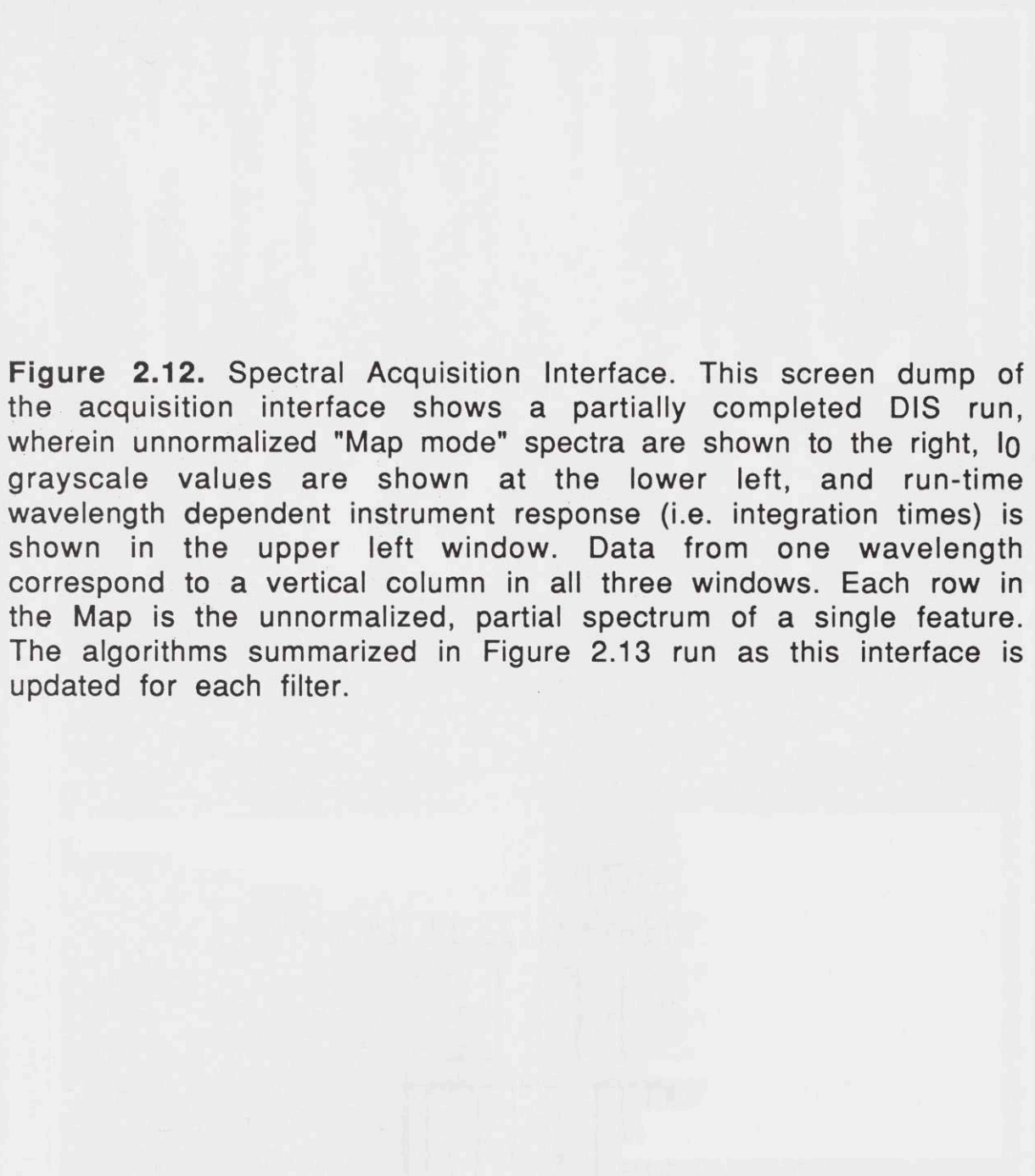
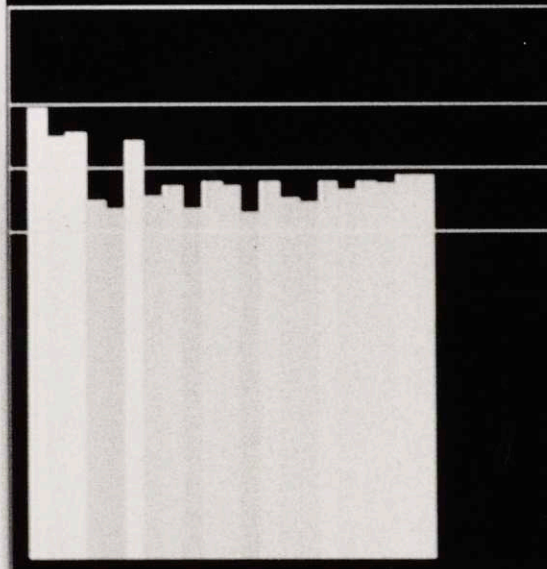
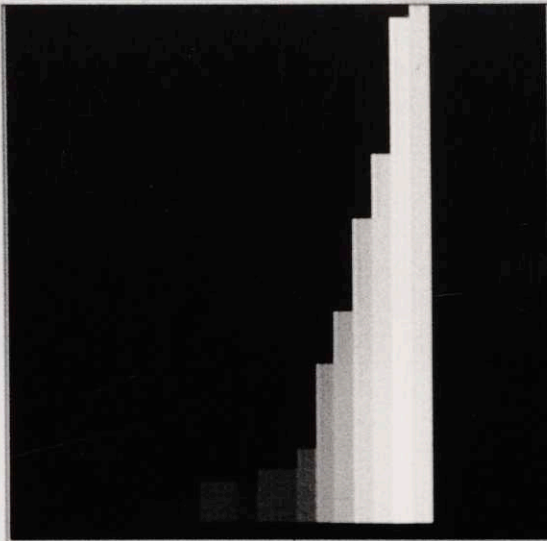


Figure 2.12. Spectral Acquisition Interface. This screen dump of the acquisition interface shows a partially completed DIS run, wherein unnormalized "Map mode" spectra are shown to the right, I₀ grayscale values are shown at the lower left, and run-time wavelength dependent instrument response (i.e. integration times) is shown in the upper left window. Data from one wavelength correspond to a vertical column in all three windows. Each row in the Map is the unnormalized, partial spectrum of a single feature. The algorithms summarized in Figure 2.13 run as this interface is updated for each filter.



47

A complete set of data for a single DIS run consists of one 12 bit image for every filter used. The camera's integration time is automatically adjusted by an efficient search algorithm until the average digitized value of the pixels in the reference region reaches the target range. Each guess is shown in the lower left of the SA display (Figure 2.12). The program can perform run-time corrections to make the search procedure more rapid. If a petri dish similar to the one currently in the instrument was last examined, then no search for the integration time will be necessary. Because the CCD camera is a linear photon counting device, the target range can be large. Adjusting the reference area to the same value for every wavelength normalizes the overall response of the system, including the wavelength sensitivity of the CCD chip and spectrum of the light source. After the target range has been reached, the image data from the reference area(s) and the colonies are stored in the computer. Finally, the integration time, reference region value, and raw spectral data are plotted in the SA interface window. After all the relevant wavelengths have been examined the 12 bit data are passed to the CaD program. This process is summarized in Figure 2.13.

2.2.5.3 Classification and Display

There are two basic windows in the CaD interface and a number of optional windows (Figure 2.14). The largest window on the right is divided into three sections: 1) The upper left section displays the spectral information in either Tile or Map mode (see below); 2) The upper right section contains a series of useful software buttons; 3) A command entry port is found at the bottom of the CaD window. The other window (lower left) is a display of the original image used for the feature extraction. One may point to a single spectrum in the Map or Tile and the corresponding object on



Figure 2.13. Spectral Acquisition Flow Chart. Consistent with user defined parameters (e.g. wavelength range) this set of algorithms is running while individual monochromatic images are recorded for each wavelength. Feature extraction has already been performed, so data are grouped by feature (e.g. each colony). Integration times are varied as a function of filter number such that a target grayscale value is achieved (within a specified range) for reference pixels.



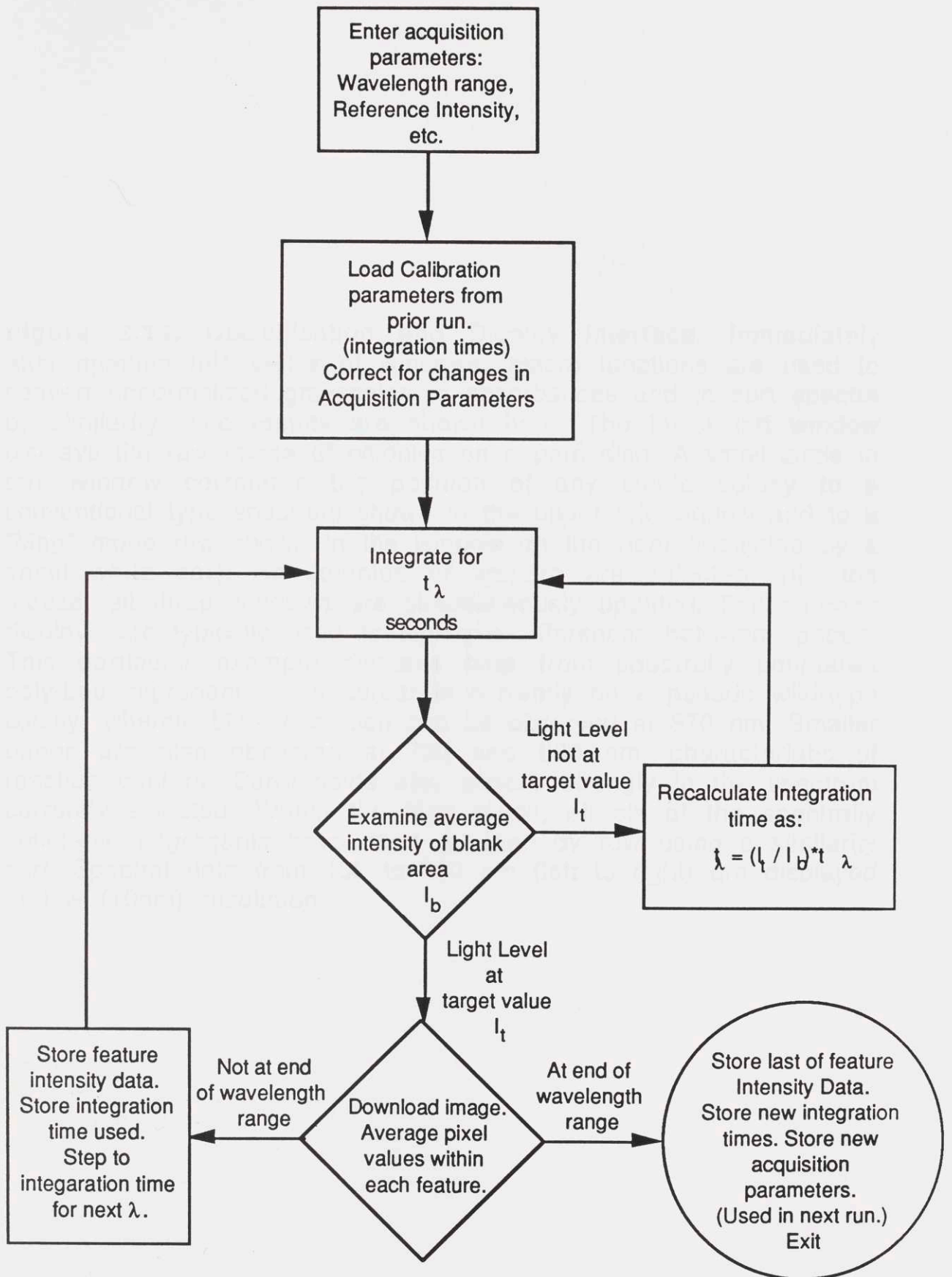
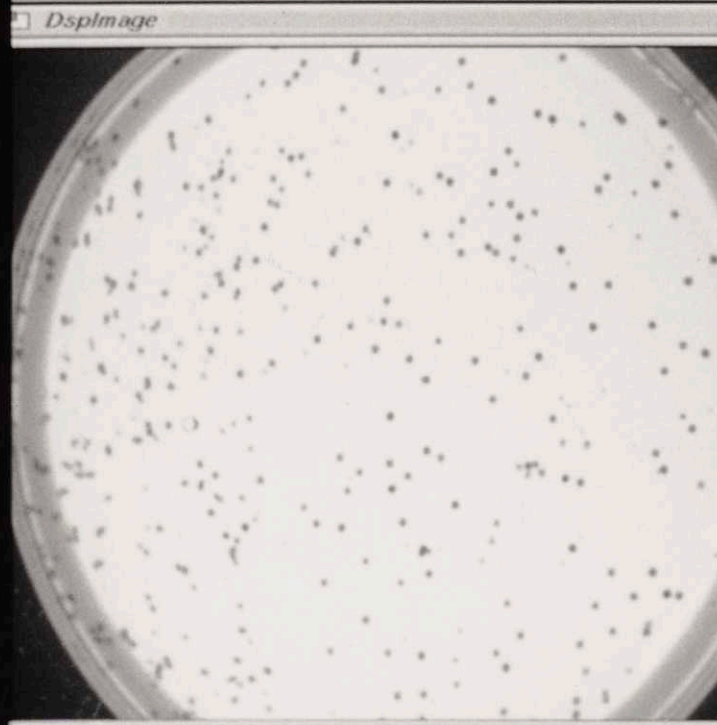
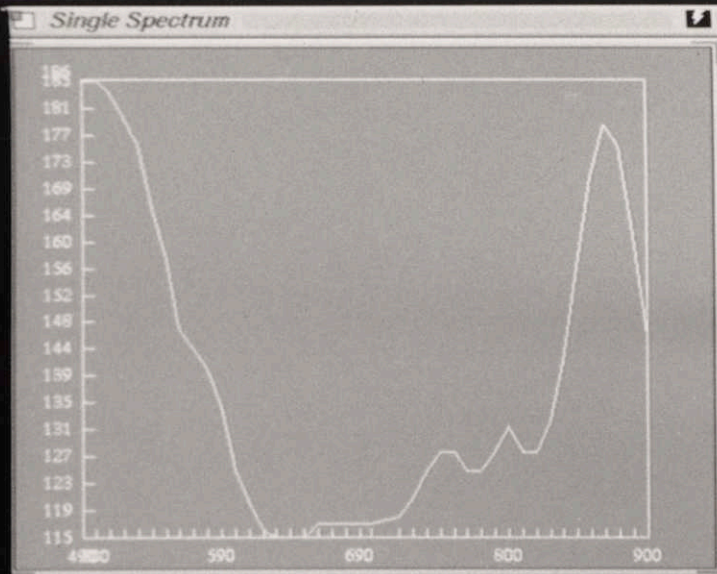


Figure 2.14. Classification and Display Interface. Immediately after opening this series of windows, macro functions are used to convert unnormalized grayscale to absorbances and to sort spectra by similarity. The results are shown here. The lower left window displays the raw image of colonies on a petri dish. A small circle in this window correlates the position of any single colony to a conventional type spectrum shown in the upper left window and to a "Map" mode row shown in the window on the right (indicated by a small white bar). As colonies or spectra are selected with the mouse, all three windows are simultaneously updated. Pseudo-color displays are typically used to highlight differences between spectra. This particular example displays data from spectrally splintered poly-Leu segregants. The cursor is currently on a pseudo wild-type colony, wherein LHI absorption can be observed at 870 nm. Smaller bands are also observed at 760 and 800 nm, characteristic of reaction centers. Carotenoids also absorb strongly in the spectrum currently selected. Within the Map mode, all six of the spectrally splintered segregants have been grouped by row using a similarity sort. Spectral data from 450 to 900 nm (left to right) are displayed at low (10nm) resolution.



Draw Test

Spectrum- 250/336 Lambda- 900-880-500

186

115

Row Up	Row Down
Column Left	Column Right
Contour	Tile
Increase X	Decrease X
Increase Y	Decrease Y
Photography	Single
Absolute Mode	Stretch Mode
Full Deflection	Quit

```

CMD> toggle
CMD> lrange 500-900
CMD>
  
```

52

the image is circled. Similarly, if one points to an object on the image, the corresponding spectrum is indicated. The upper left window is continuously updated in real-time to display a conventional spectrum for the currently selected colony.

A ground state absorption spectrum can be calculated using the Beer-Lambert law:

$$A = \log_{10}(I_0/I) \quad \text{Eq. 2.2}$$

where I_0 is the reference intensity, determined by the average value of the reference area pixels at each wavelength and I is the object intensity calculated by the average value of pixels within the feature's perimeter. Grayscale values can be entered directly into this equation because the CCD is a linear photon counter.

The spectra may be displayed in two modes. The Tile mode (Figure 2.15) draws each spectrum as a single graph within a box. Each box may be rescaled as desired. Figure 2.16 demonstrates the second mode which is called Map mode. Here, each colony spectrum is shown as a horizontal line across the window; each point on the line corresponds to a specific wavelength. The color (remapped to a grayscale for this publication) at each point in the line is determined by the absorption value at the corresponding wavelength (white for high absorption, black for zero absorption). Map mode allows for very rapid identification of spectral class (demonstrated below). In any mode, the spectra may be scaled either on: a) an absolute scale, where the spectra are plotted relative to the maximum $[\log_{10}(4095/1)]$ and minimum $[\log_{10}(4095/4096)]$ absorbances measurable by the spectrometer; b) a scale relative to the lowest and highest absorbances found anywhere in the image, or c) "full-deflection" where each spectrum is scaled



Figure 2.15. Tile Mode. As an alternative to Map mode, the Tile mode can be used to display all of the colony spectra from a petri dish. The Tile mode normally appears as an option within the CaD user interface. This figure shows a screen dump of 81 poly-Leu segregants. All spectra are from 450 to 900 nm (left to right). Tiles in the lower row are similar to the pseudo wild-type segregant displayed in Figure 2.14. The roughness of the spectra is an artifact of data transfer to a laser printer.



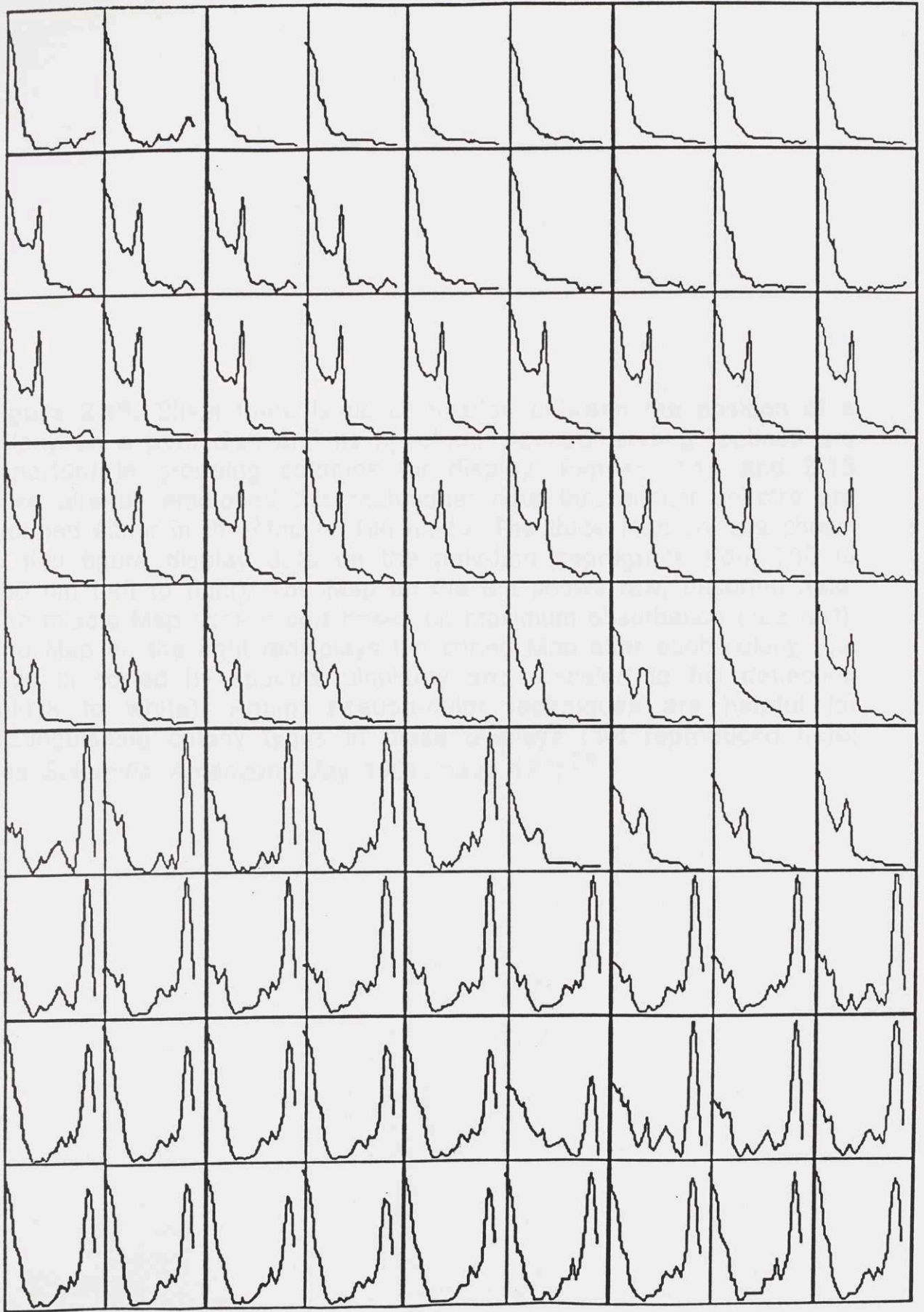
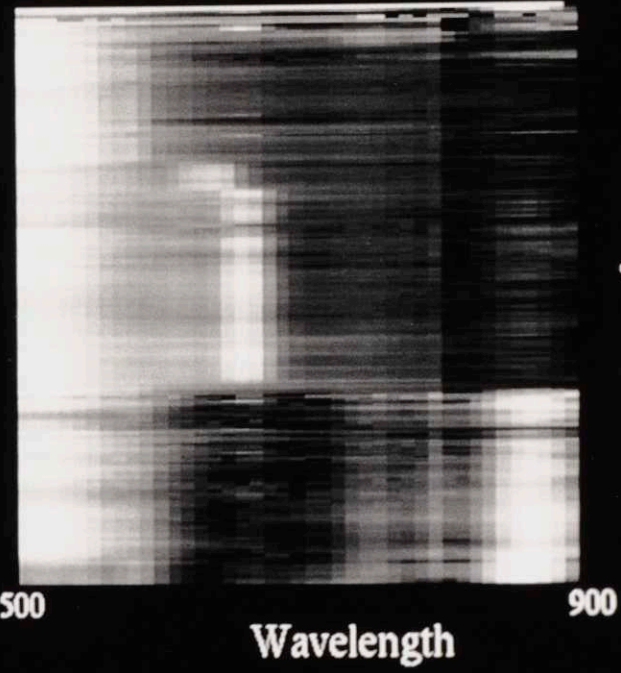
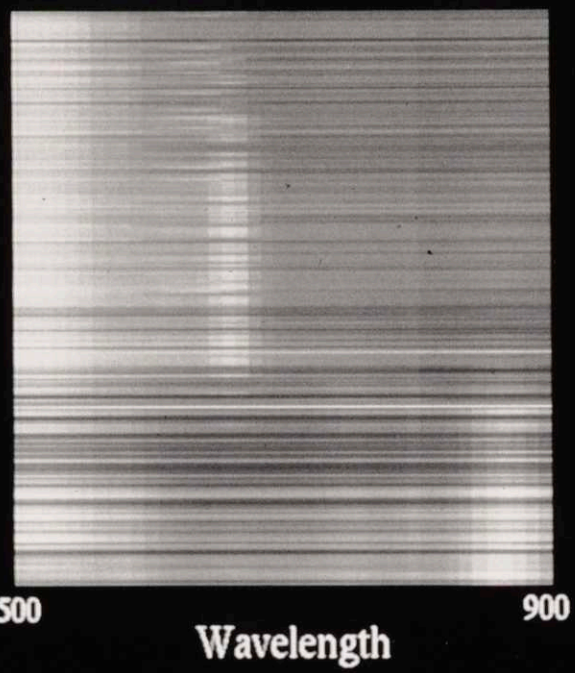
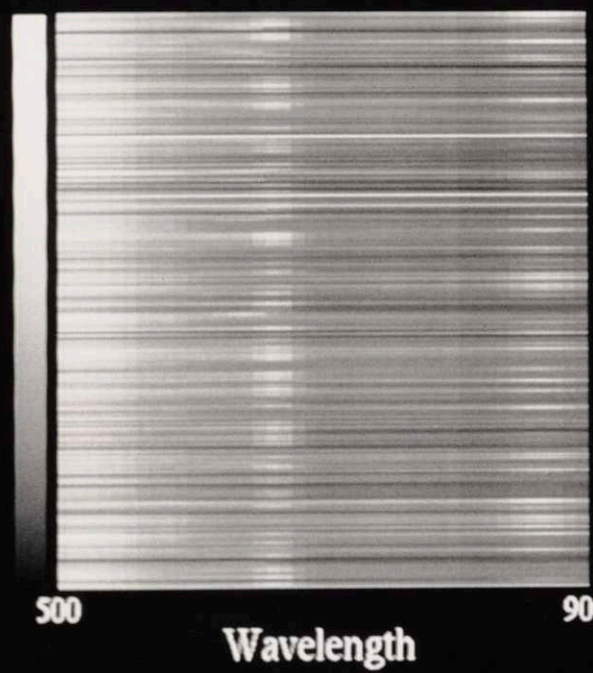


Figure 2.16. Since there is no correlation between the position of a colony on a petri dish and its spectrum, spectral sorting routines are important in grouping colonies for display. Figures 2.14 and 2.15 have already employed this technique: note that similar spectra are grouped either in the Map or Tile mode. The three Map spectra shown in this figure display data on the poly-Leu segregants from 500 to 900 nm (left to right). The Map on the left shows raw, unsorted data. The middle Map uses a sort based on maximum absorbance (see text). The Map on the right redisplay the sorted Map after each colony (i.e. row) is sorted by spectral similarity and rescaled to full deflection (black to white). Again, pseudo-color techniques are helpful for distinguishing colony types in these displays (not reproduced here; see *Scientific American*, May 1991, page 123).²⁹

57



between its own maximum and minimum absorption. These display modes may affect the behavior of the classification algorithms.

Once the spectra have been calculated they are generally in a disordered ensemble (Map mode) as shown in Figure 2.16a. These spectra are from a petri dish containing the results of the mutagenesis experiment described above (i.e. replating the poly-Leu mutant). The "venetian blind" effect occurs because there are many different types of colonies and each type is probably next to a dissimilar segregant. To order the Map and classify each spectrum, spectra are sorted from colonies of highest absorbance to those of lowest absorbance (Figure 2.16b). This type of sort can be limited to particular wavelengths. Next, a similarity sort is performed. The action of this sort depends strongly on the specified wavelength range and on the scaling mode of the data. Generally, the full-deflection mode is employed. The similarity sort essentially calculates the sum of squares of differences (SSD) between a given spectrum and each of the remaining unsorted spectra. The unsorted spectrum producing the smallest SSD is then placed directly above the current spectrum. This procedure is then repeated for the newly placed spectrum until all the spectra have been sorted. Figure 2.16c displays the results of such a similarity sort.

These are only two of a number of sorts which may be performed. Other sorts can be based on: peak ratios, wavelength of maximum absorbance, peak widths, colony size, etc. Spectra can be added, subtracted, divided and averaged. Spectral ensembles from different runs may be combined and edited.

2.2.6 System Development

Each component of the DIS system may be improved to some extent. The laws of physics place definite limits on the precision of the system: photon shot noise will always be a problem. Narrow bandwidths are inconsistent with counting more photons, so improved differential sensitivity and improved wavelength resolution involve competing factors. However, practical improvements in the DIS are still feasible by minimizing the time expended in analyzing a given number of mutants. The current generation DIS is capable of processing on the order of 10^4 mutants per day. We expect that this can be increased by one order of magnitude if, for example, the system is fully automated and memory mapped cameras are employed.

Camera parameters that may be altered to improve system performance are: 1) camera shutter speed, 2) CCD readout and data transfer rate, and 3) CCD array size. Currently, the shutter times may be set with 0.1 second resolution. However, faster shutters (0.01 second) are commercially available. Using a faster shutter, target grayscale levels could be obtained in 1/10th the time (with the same precision as currently possible) by opening the lens two f-stops. Furthermore, readout rates 25 times faster than the current IEEE 488 interface (20k pixels/second) can be achieved by using a memory mapped VME interface to the CCD controller. Increasing the size of the CCD chip will obviously increase the readout time, but if a 2048 x 2048 format chip is used, PSN is reduced by $(4,000,000 / 200,00)^{0.5}$ or about four times. Digitization to 16 bits would be required to see this improvement, since the present level of PSN is approximately one grayscale unit on a 12 bit scale.

2.2.8 Acknowledgements

The light source currently has a maximum resolution of 5 nm and a throughput of approximately 2.5 mW/cm² per nanometer. At room temperature, a resolution of 1 nm is more than adequate to obtain accurate spectra of vibrationally broadened substances. This additional resolution might be achieved through tilting the Fabry-Perot filters.²⁸ Further, our system currently operates efficiently over the spectral range from 930 to 400 nm. Extension of this range into the UV would allow the spectrometer to detect a wider range of organic compounds which may be produced and/or secreted by microorganisms. In order to extend the spectral range of the DIS into the UV all optics would have to be replaced with either silica or quartz.

2.2.7 Conclusions

Any system which relies on the identification of colored substances on a two dimensional surface may be amenable to digital imaging spectroscopy. DIS is an excellent tool for obtaining *in vivo* colony spectra. Over 1000 spectra may be obtained simultaneously at a sensitivity greater than or equal to a conventional spectrometer. This makes the DIS an excellent device for screening combinatorial cassette libraries, providing that an intrinsic or extrinsic colorimetric assay is available. Recently, DIS spectra of sperm whale myoglobin and cytochrome *b5* have been obtained in *E. coli*. (Unpublished results) Both these proteins have absorption in the heme Soret band around 400-420 nm. In the context of this thesis, the intrinsic pigments of the RC and LH antennae are amenable to DIS analyses for a variety of protein structure and function problems.

2.2.8 Acknowledgements

This work was supported by NIH GM42645, DOE DE-FG02-90ER20019, DOE URI DE-FG05-91ER79031, and by the Human Frontiers Science Program. A.P.A. was also supported by an NIH Biophysics Training Grant. We thank Christine A. Goddard for technical assistance.

2.3 Epilogue

When the full implementation of DIS was initially conceived, it was designed to be able to rapidly screen the absorption spectra of thousands of colonies of *Rb. capsulatus*. This would only be interesting, of course, if there were great variability in the spectra from colony to colony. Obviously, if this variability occurred when all the colonies were expressing the same protein the data would be very hard to analyze and we would have to focus on the metabolic differences due to the growth conditions under which a colony developed. Luckily and expectedly, colonies expressing the same proteins had essentially identical spectra.

Other research in the laboratory focussed on producing large, diverse populations of proteins using modifications of combinatorial cassette mutagenesis (see Chapters 3-6). Application of these techniques to the photosynthetic proteins (now being performed in the laboratory) results in bacterial colonies whose spectra are determined by which of the population of proteins they express. DIS is a perfect device for screening these millions of colony types. However, as mentioned above, DIS can currently screen up to only 10,000 colonies a day (i.e. about 10-20 plates). Automating the system, along with the other improvements discussed above, will do much to increase the throughput and decrease the tedium involved in processing DIS data. The first major

step towards this goal would be to automate feature extraction. Approximately one third of the time needed to obtain a DIS spectrum is dedicated to the binarization of the petri dish image and editing the resultant picture to reject extraneous noise and separate "merged" colonies. An algorithm, the Binarization Algorithm with Hysteresis (BAH), is now under development to perform these operations automatically. Figure 2.17 shows the basic concept behind the process.

Figure 2.18 shows the result of operating on a petri dish image. Once this algorithm is optimized it might be used as part of an overall robotic system in which many petri dishes are automatically examined without the need for an operator being present. be separated.

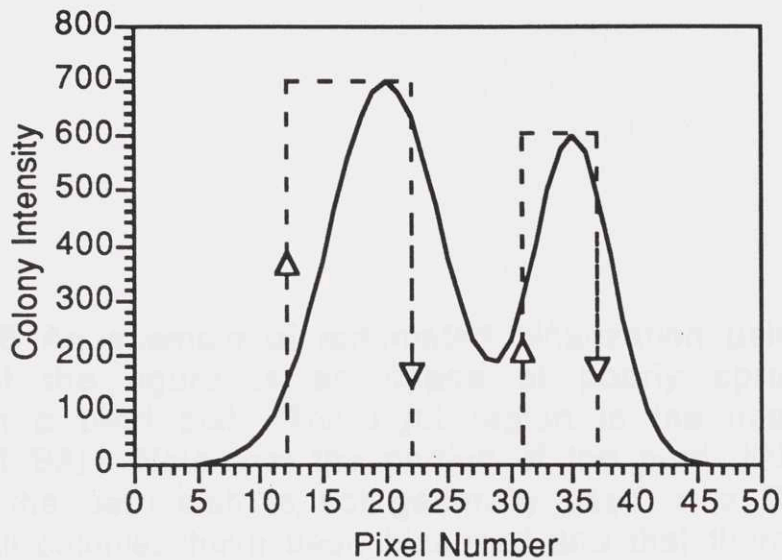
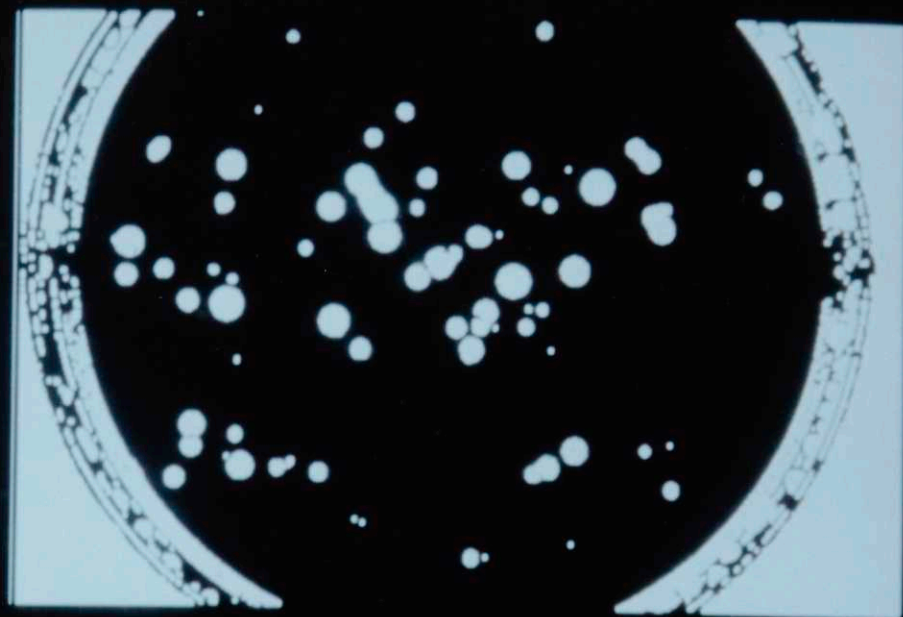
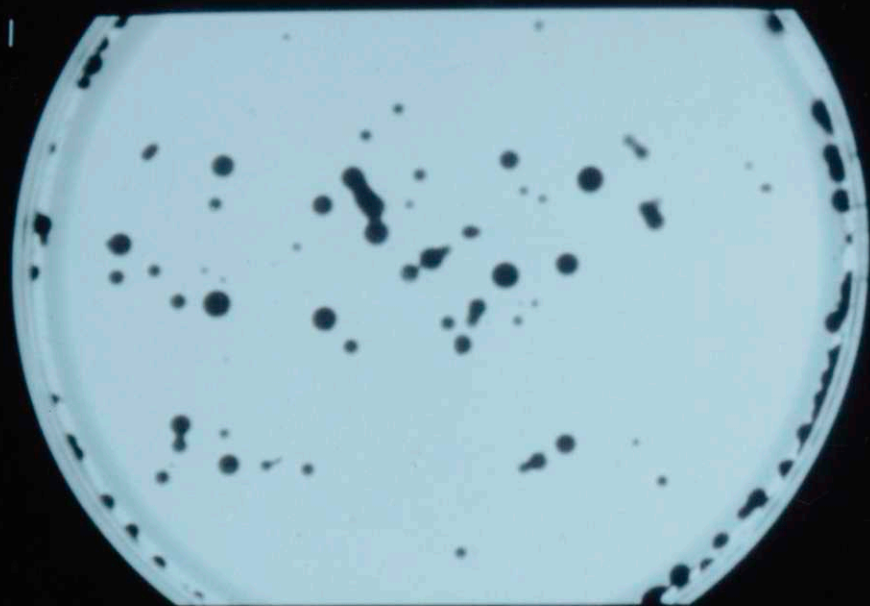


Figure 2.17 One trace from the Binarization Algorithm with Hysteresis (BAH). A single colony is assumed to have a symmetric, singly-peaked shape. Thus, the smooth, double-humped graph represents a densitometric trace through an image of two colonies (centered at pixel numbers 20 and 35) which have grown into one another. The dotted trace superimposed on the graph is a representative of a left-to-right sweep of BAH. When this trace is non-zero, a white pixel is written to a temporary image otherwise a black pixel is written. The single parameter for this procedure is the hysteretic width (HW). In this case, HW is set to 50. When a positive intensity change of HW between two pixels is encountered, the algorithm switches "on" (returns a non-zero value.) Then, when a change of $-HW$ from the maximum intensity observed after switching on is obtained, the algorithm switches "off" (returns a zero value). This process is performed both from left-to-right and right-to-left for every line in the image. Another temporary image is created from traces run from top-to-bottom and bottom-to-top. The two resultant images are then AND-ed together to produce the final binary image. The hysteretic effect will cause many types of merged colonies to be separated. Figure 2.18 shows the effects of this algorithm on a pathological image of a petri dish.

Figure 2.18 An example of automated binarization using BAH. The left half of the figure is an image of poorly spread bacterial colonies on a petri dish. The right region is the image after the operation of BAH. Note that the portion of the petri dish outside the the rim of the petri dish is not generally used in a DIS spectrum. Note that all colonies have been binarized and that there are at least two examples where connected colonies were separated. Further optimization of the hysteretic width (see caption to Figure 2.17) in combination with standard separation techniques such as shrinking and expanding, convolution, etc. should yield reproducibly edited, binary images suitable for automated feature extraction. The wavelength under which the original image is obtained is also an important consideration. It should be chosen such that colony profiles are as steep as possible.



2.4 References

1. Yang, M. M. and D. C. Youvan. 1988. Applications of Imaging Spectroscopy in Molecular Biology: I. Screening Photosynthetic Bacteria. *Bio/Technology* **6**: 939-942.
2. Coleman, W. J. and D. C. Youvan. 1990. Spectroscopic Analysis of Genetically Modified Photosynthetic Reaction Centers. *Annu. Rev. Biophys. Biophys. Chem.* **19**: 333-367.
3. Miller, J. H. 1972. *Experiments in Molecular Genetics*. Cold Spring Harbor Laboratory, Cold Spring Harbor.
4. Arkin, A. P., E. Goldman, S. J. Robles, W. J. Coleman, C. A. Goddard, M. M. Yang and D. C. Youvan. 1990. Applications of Imaging Spectroscopy in Molecular Biology II: Colony Screening Based on Absorption Spectra. *Bio/Technology* **8**: 746-749.
5. Matteucci, M. D. and H. L. Heyneker. 1983. Targeted Random Mutagenesis: The Use of Ambiguously Synthesized Oligonucleotides to Mutagenize Sequences Immediately 5' of an ATG Initiation Codon. *Nucl. Acids Res.* **11**: 3113-3121.
6. Reidhaar-Olson, J. F. and R. T. Sauer. 1988. Combinatorial Cassette Mutagenesis as a Probe of the Informational Content of Protein Sequences. *Science* **241**: 53-57.
7. Reidhaar-Olson, J. F., J. U. Bowie, R. M. Breyer, J. C. Hu, K. L. Knight, W. A. Lim, M. C. Mossing, D. A. Parsell, K. R. Shoemaker and R. T. Sauer. 1991. Random Mutagenesis of Protein Sequences Using Oligonucleotide Cassettes. *Methods in Enzymology* **208**: 564-587.
8. Hermes, J. D., S. C. Blacklow and J. R. Knowles. 1990. Searching Sequence Space by Definable Random Mutagenesis- Improving the Catalytic Potency of an Enzyme. *Proc. Natl. Acad. Sci. USA* **87**: 696-700.
9. Smith, G. P. 1985. Filamentous Fusion Phage: Novel Expression Vectors That Display Cloned Antigens on the Virion Surface. *Science* **228**: 1315-1317.

10. Arkin, A. P. and D. C. Youvan. 1992. Optimizing Nucleotide Mixtures to Encode Specific Subsets of Amino Acids for Semi-Random Mutagenesis. *Bio/Technology* **10**: 297-300.
11. Arkin, A. P. and D. C. Youvan. 1992. Application of a Novel Technique to Protein Engineering: Simulation of Recursive Ensemble Mutagenesis. *Proc. Natl. Acad. Sci.* submitted.
12. Goldman, E. R. and D. C. Youvan. 1992. Combinatorial Mutagenesis of the Light Harvesting II Antennae Using Intelligent Dopes and Digital Imaging Spectroscopy. In Preparation.
13. Robles, S. J. and D. C. Youvan. 1992. Combinatorial Mutagenesis of the Photosynthetic Reaction Center. In Preparation.
14. Youvan, D. C. 1991. Photosynthetic Reaction Centers: Interfacing Molecular Genetics and Optical Spectroscopy. *TIBS* **16**: 145-149.
15. Gonzales, R. C. and P. Wintz. 1987. *Digital Image Processing*. Addison-Wesley Publishing Co., inc., Reading.
16. Youvan, D. C., E. J. Bylina, M. Alberti, H. Begusch and J. E. Hearst. 1984. Nucleotide and Deduced Polypeptide Sequence of the Photosynthetic Reaction Center, B870 Antenna, and Flanking Polypeptides from *Rps. capsulata*. *Cell* **37**: 949-957.
17. Zuber, H. 1986. Structure of Light Harvesting Antenna Complexes of Photosynthetic Bacteria, Cyanobacteria and Red Algae. *TIBS* **11**: 414-419.
18. Bylina, E. J. and D. C. Youvan. 1989. Photosynthesis in *Rhodospirillaceae*. In *Genetics of Bacterial Diversity*. Hopwood and Chater. (Ed.) (Academic Press, New York) 87-106.
19. Zuber, H. 1990. Consideration on the Structural Principles of the Antenna Complexes of Phototrophic bacteria. In *Molecular Biology of Membrane-Bound Complexes in Phototrophic Bacteria*. G. Drews and E. A. Dawes. (Ed.) (Plenum Press, New York) 161-180.
20. Sambrook, J., E. F. Fritsch and T. Maniatis. 1989. *Molecular Cloning: A Laboratory Manual*. Cold Spring Harbor Laboratory Press, New York.

21. Youvan, D. C., S. Ismail and E. J. Bylina. 1985. Chromosomal Deletion and Plasmid Complementation of the Photosynthetic Reaction Center and Light-Harvesting Genes. *Gene* **38**: 19-30.
22. Bylina, E. J., S. Ismail and D. C. Youvan. 1986. Plasmid pU29, A Vehicle for Mutagenesis of the Photosynthetic *puf* Operon in *Rhodopseudomonas capsulata*. *Plasmid* **16**: 175-181.
23. Marrs, B. L. 1978. Genetics and Bacteriophage. *In* The Photosynthetic Bacteria. R. K. Clayton and W. R. Sistrom. (Ed.) (Plenum Publishing Corporation, New York) 873-883.
24. Kolaczowski, S. V., E. J. Bylina, D. C. Youvan and J. R. Norris. 1990. Examination of the *Rhodobacter capsulatus* Special Pair in Wild-Type and Heterodimer-Containing Reaction Centers by Time-Resolved Optically Detected Magnetic Resonance. *In* The Molecular Biology of Membrane-Bound Complexes in Photosynthetic Bacteria. G. Drews. (Ed.) (Plenum Press, New York) 305-312.
25. Epperson, P. M., J. V. Sweedler, M. B. Denton and G. R. Sims. 1987. Electro-optical Characterization of the Tektronix TK512M-011 Charge-Coupled Device. *Opt. Eng.* **26**: 715-724.
26. Photometrics. 1989. Charge-Coupled Devices for Quantitative Electronic Imaging. Photometrics Ltd.
27. Goebel, D. G. 1967. Generalized Integrating Sphere Theory. *Appl. Opt.* **6**: 125-128.
28. Macleod, H. A. 1986. Thin-film Optical Filters. MacMillan Publishing Company, New York.
29. Stix, G. 1991. Protein Probe: Remote-Sensing Technique Screens Bacterial Cultures. *Scientific American*. 123.

Chapter 3

Addressing Complexity

Optimized Nucleotide Mixtures

Historically, the development of the following procedure, a combination of optimized nucleotide mixtures, was based on the use of random DNA libraries. However, it is now possible to design DNA libraries with specific properties. In this section, we describe a method for the design of DNA libraries with specific properties. The design of DNA libraries with specific properties is a complex task. It involves the selection of nucleotide sequences that are likely to be functional and the optimization of the library composition. This process is often iterative and can be challenging. However, the use of optimized nucleotide mixtures can significantly improve the efficiency of the library design process. In this section, we describe a method for the design of DNA libraries with specific properties. The design of DNA libraries with specific properties is a complex task. It involves the selection of nucleotide sequences that are likely to be functional and the optimization of the library composition. This process is often iterative and can be challenging. However, the use of optimized nucleotide mixtures can significantly improve the efficiency of the library design process.

The design of DNA libraries with specific properties is a complex task. It involves the selection of nucleotide sequences that are likely to be functional and the optimization of the library composition. This process is often iterative and can be challenging. However, the use of optimized nucleotide mixtures can significantly improve the efficiency of the library design process.

Figure 3.1

The design of DNA libraries with specific properties is a complex task. It involves the selection of nucleotide sequences that are likely to be functional and the optimization of the library composition. This process is often iterative and can be challenging. However, the use of optimized nucleotide mixtures can significantly improve the efficiency of the library design process.

3.1 Reducing Complexity

Historically, the development of the following procedures, a modification of traditional combinatorial cassette mutagenesis (CCM) came after the material in Chapter 4. However, it may be argued that the material presented here logically comes earlier. In CCM, many sites in a protein are mutagenized at once and in a random fashion in order to produce a library of proteins which may exhibit novel properties. (Throughout the following chapters we also use the terms "population" and "ensemble" for diverse sets of proteins) The complexity of these libraries grows exponentially with the number of sites being mutagenized, thus, the ability to screen all the members in the library rapidly diminishes. As early as 1987¹, thought was being given to methods of reducing the number of amino acids expressed at each mutagenic site, and thus the library complexity, by exploiting properties of the genetic code. The search for intelligent ways of limiting the amino acid sets one wished to be present at each site lead to the following paper² and to the methods presented in Chapters 4 and 6.

3.2 Optimized Nucleotide Mixtures for Combinatorial Cassette Mutagenesis

3.2.1 Summary

In random mutagenesis, synthesis of an **NNN** triplet (i.e. equiprobable **A, C, G,** and **T** at each of the three positions in the codon) could be considered an optimal nucleotide mixture because all 20 amino acids are encoded. **NN(G,C)** might be considered a slightly more intelligent "dope" because the entire set of amino acids is still encoded using only half as many codons. Using a general algorithm

described herein, it is possible to formulate more complex doping schemes which encode specific subsets of the twenty amino acids, excluding others from the mix. Maximizing the equiprobability of amino acid residues contributing to such a subset is suggested as an optimal basis for performing semi-random mutagenesis. This is important for reducing the nucleotide complexity of combinatorial cassettes so that "sequence space" can be searched more efficiently. Computer programs have been developed to provide tables of optimized dopes compatible with automated DNA synthesizers.

3.2.1 CCM to Search Sequence Space

Protein design aided by molecular genetics has progressed towards very complex mutagenesis schemes.³⁻⁶ This complexity arises because of an inability to accurately predict the structure and function of proteins from their primary amino acid sequences.⁷ Techniques such as combinatorial cassette mutagenesis (CCM) explore large numbers of mutations in a protein.⁸⁻¹⁴ However, even for a small subset of sequence space, the complexity of the protein library resulting from random CCM is too large to be completely screened for functional proteins. For example, mutagenesis of 12 amino acids using NNN triplets results in a DNA complexity of 4^3 and a protein complexity of 20^{12} . Furthermore, because different codons may translate to the same amino acid, each protein sequence does not appear equiprobably in the CCM library. In this example, poly-leucine would appear 6^{12} times more frequently than poly-methionine.

To increase screening efficiency, it is desirable to reduce the total number of proteins in the CCM library and to make the probability of observing each protein equal. In addition to reducing the complexity, one would also like to minimize the number of

proteins which are nonfunctional. As a first approximation, the search can be limited to sequences related to wild-type in some space of physical parameters or by phylogeny. In the latter case, an optimized doping scheme would be calculated for each position mutagenized using a subset of the twenty amino acids. In addition, the doping scheme would be optimized for equiprobable expression of each amino acid within the subset. We define an optimized dope as a nucleotide mixture which maximizes the probability of observing the entire subset of desired amino acids and expresses the members of this subset equiprobably.

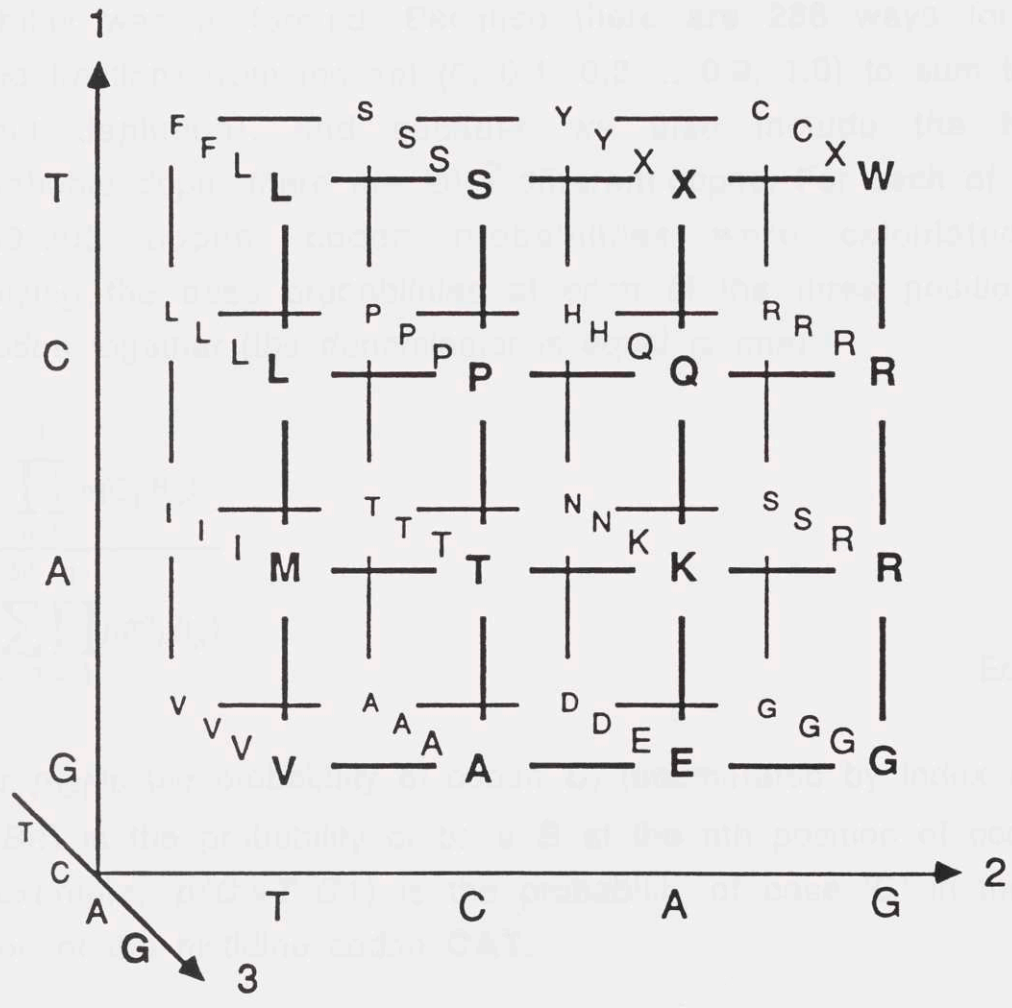
Consider the grid-like representation of the genetic code shown in Figure 3.1. For certain subsets of amino acids, a doping scheme can be developed by inspection. For example, it is possible to construct a dope that includes all three aromatic residues (F, Y, and, W) while minimizing all other amino acids. We proceed by circling all occurrences of these three residues in Figure 3.1. A pattern is immediately obvious in that all occurrences of codons encoding the aromatic amino acids fall in the top "plane" of the code which means that only **T** is needed in the first position of the codon. Next, we draw a line through all four serine codons to indicate that **C** is to be avoided in the second position. An **A** in the third position is also avoided. This yields the dope $[T(\mathbf{A},\mathbf{G},\mathbf{T})(\mathbf{C},\mathbf{G},\mathbf{T})]$ which includes 2 Phe, 1 Leu, 2 Tyr, 1 Trp, 1 stop, and 2 Cys codons. Further inspection shows that **C** can also be omitted from the third position, such that $[T(\mathbf{A},\mathbf{G},\mathbf{T})(\mathbf{G},\mathbf{T})]$ encodes 1 Phe, 1 Leu, 1 Tyr, 1 Trp, 1 stop, and 1 Cys codon. This latter dope might be judged optimal because the three aromatic residues occur equiprobably. However, the probability of observing all the desired amino acids decreases from 55.5% to 50%. Generally, our definition of "optimal" prescribes choosing the first dope over the second. Below, we describe an algorithmic approach to

Figure 3.1. Three-dimensional representation of the genetic code using the single letter designation for amino acids. Axes 1, 2, and 3 correspond to the first, second, and third positions in the codon.

This problem is amenable to computer calculation at high resolution.

3.2.2 Results and Utility

Robot utility is defined as the number of robot hours required to produce one unit of output. The utility of a robot is a function of its speed and the amount of time it spends on each task.



Eq. 3.1

Eq. 3.2

this problem that is amenable to computer calculation at high resolution.

3.2.2 Results and Discussion

An exhaustive computer search of all possible single codon dopes with a fractional resolution of 0.1 (10%) in nucleotide probability was performed. Because there are 286 ways for four ordered fractions from the set (0, 0.1, 0.2 ... 0.9, 1.0) to sum to 1.0 (without depletion), and because we also include the **NNN** equiprobable dope, there are 287^3 different dopes. For each of these 23,639,903 dopes, codon probabilities were calculated by multiplying the base probabilities at each of the three positions in the codon together (the denominator is equal to one):

$$p_{C_i} = \frac{\prod_{n=1}^3 p(C_i, B_n)}{64^3 \sum_{i=1}^{64^3} \prod_{n=1}^3 p(C_i, B_n)} \quad \text{Eq. 3.1}$$

where p_{C_i} is the probability of codon C_i (enumerated by index i) and $p(C_i, B_n)$ is the probability of base B at the n th position of codon i . For example, $p(\mathbf{CAT}, C1)$ is the probability of base 'C' in the first position of the histidine codon **CAT**.

The probability of a specific amino acid (p_A) was calculated by summing the probabilities of all codons for that amino acid:

$$p_A = \frac{\sum_{i=\{\text{codons for A}\}} p_{C_i}}{\sum_{i=\{\text{all codons}\}} p_{C_i}} \quad \text{Eq. 3.2}$$

The average physical properties of the CCM library of amino acids can be calculated by weighting physical parameters, for example:

$$H = \sum_{\{\text{all acids}\}} p_A H_A$$

$$M = \sum_{\{\text{all acids}\}} p_A M_A$$

Eqs. 3.3, 3.4

where H and M are the average hydrophathy¹⁵ and molar volume¹⁶ of residues encoded by the dope, respectively, and H_A and M_A are the actual hydrophathy and molar volume values of each acid. In this study, the molar volumes are the apparent molar volumes of the amino acid residues as calculated in reference 16. Note that the p_{AS} used in these last two equations are renormalized to exclude the probabilities of stop codons.

We have found that the equation for group probability (p_G):

$$p_G = \prod_{\{\text{subset acids}\}} p_A$$

Eq. 3.5

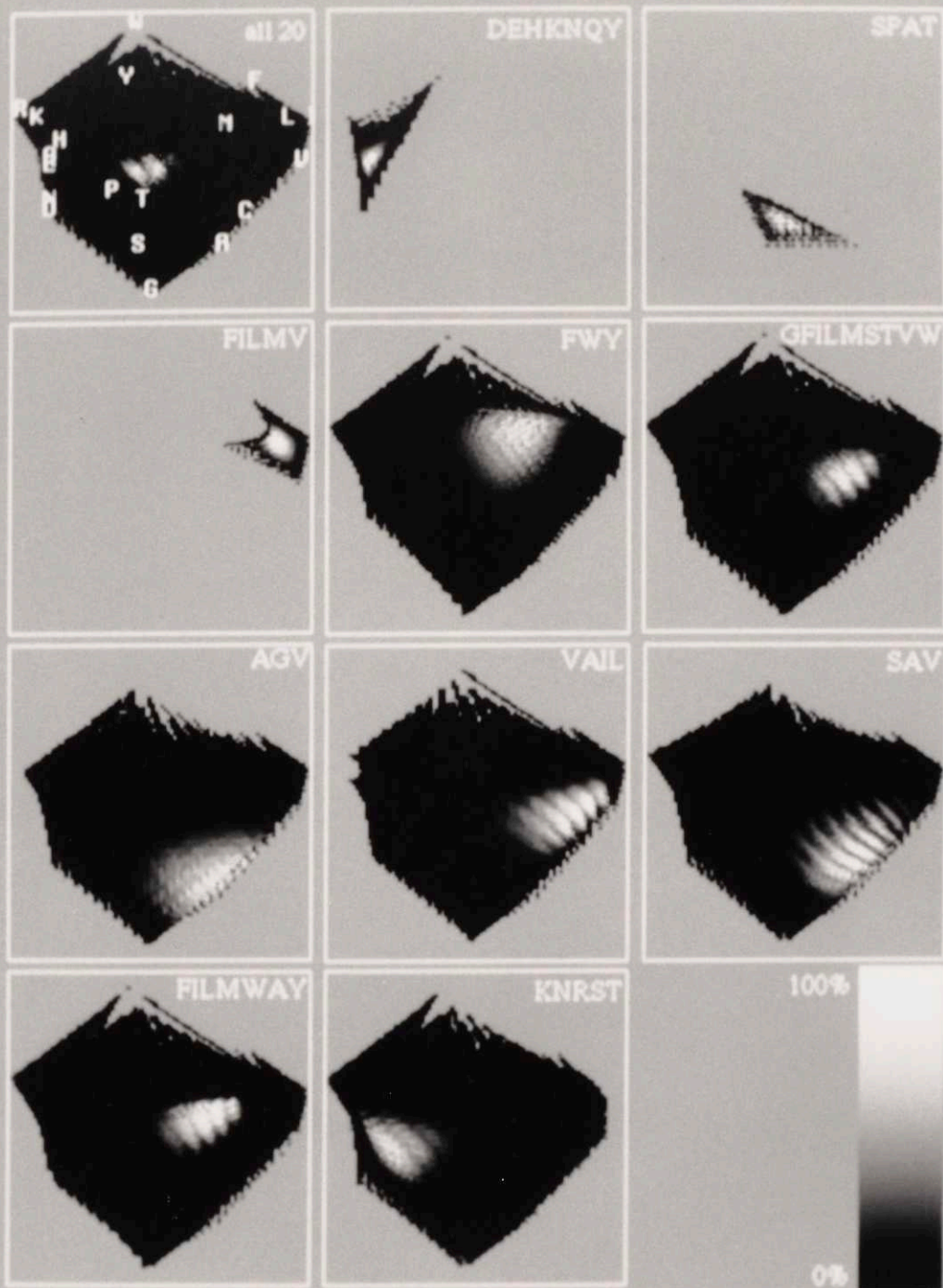
where the product is taken over all the amino acids in the targeted subset has the desirable properties of: 1) directly measuring the probability of observing all the amino acids in the target subset, 2) producing a maximum value when all amino acids in the subset are equiprobable and the sum of their probabilities is equal to one (proof given below), 3) generating a value of zero when any amino acid in the subset has a probability of zero, and 4) remaining defined for all possible subsets of amino acids. The properties of the group probability equation are highly analogous to the simple inspection

technique described in the introduction. Alternative equations, such as for Shannon entropy¹⁷ (SE) or the sum of square of differences (SSD) are pathological with respect to at least one of these stated points. For example, SE is not defined if a subset of the total alphabet is used. SE and SSD equations tend to zero certain members of the alphabet and regain favorable values by compensatory changes in the other probabilities (data not shown).

Group probabilities (p_G) were calculated for eleven different subsets of targeted amino acid residues. Data are displayed as gray scale maps in hydrophathy-molar volume space (H-M space) in Figure 3.2. This plane was chosen because recent studies, both theoretical and experimental, indicate that hydrophathy and molar volume are perhaps the most important factors governing the packing of a protein.^{9, 17-20} These maps show that there is a distribution of dopes around the calculated optimum which have nonzero group probability. These suboptimal dopes might be examined by an expert for specific properties (e.g. the minimization of the probability of an unwanted amino acid.)

For the group probability (p_G) panels displayed in Fig. 2, the physical parameter planes are composed of approximately 10,000 bins (100 divisions per axis). The maximum p_G value falling within each bin is displayed in the figure by ramping the gray scale according to the bar at the lower right. Gray values range from black (minimum p_G is zero) to white (maximum p_G for that panel). At this resolution of doping and binning, some portions of H-M space are inaccessible using any combination of amino acids (background gray areas). Certain areas of H-M space, such as the one containing tryptophan, can only be reached using one specific codon.

Figure 3.2. High resolution contour maps of group probabilities (p_G) displayed in hydrophathy-molar volume space. Group probabilities within each panel are scaled from the lowest value (zero = black) to the highest value (maximum p_G = white) with intermediate p_G values scaled according to the map shown in the lower right panel. The upper left panel (labeled "all 20") displays results from high resolution calculations involving **NNN** dopes (see text). All twenty amino acids are labeled in this panel; (EQ) and (DN) are over-struck because of similar H-M values. The next three panels display three separate calculations based on fixing the second position of the codon. Amino acids DEHKNQY are encoded by **NAN**; SPAT are encoded by **NCN**; FILMV are encoded by **NTN**. Amino acids R, W, C, and G lie outside of any of these three contours because **NGN** shows no correlation in H-M space. The remaining panels show seven examples of different subsets of amino acid residues pooled according to the phylogenetic example given in the text. All data displayed in this figure have been calculated at high resolution, as compared to the low resolution data presented in Table 3.1.



The upper-left panel in the Figure 3.2 shows a maximum group probability map for all twenty amino acids. This plot indicates that the best dopes are found in the center of the plane (as expected), because codons similar to **NNN** produce average values for hydrophathy and molar volume and have a non-zero group probability. In this map, p_G values decrease sharply away from the center because if any one of the 20 p_A values equals zero, then p_G is zero. Table 3.1 lists data related to the maximum p_G values in this plot. The tabular data are simpler than the plotted data because only 15^3 possible dopes are analyzed (synthesizer resolution). These correspond to dopes that are readily made on a commercial DNA synthesizer: 25% (each) **A,C,G**, and **T**, or 33% of any three nucleotides, or 50% of any two nucleotides, or 100% **A, C, G**, or **T**. This totals $1 + 4 + 6 + 4 = 15$ different mixes per each position of the codon. The higher resolution data displayed in Figure 3.2 can be recalled from extensive computer files.

The next three panels in Figure 3.2 display data from three separate calculations involving **NAN**, **NCN**, and **NTN** which encode D,E,H,K,N,Q,Y, and S,P,A,T, and F,I,L,M,V (single letter code), respectively. These particular dopes were chosen because the second codon position is a direct determinant of the hydrophathy and molar volume of the encoded amino acid residue.²¹⁻²³ An '**A**' in the middle position codes for hydrophilic residues; '**C**' codes for small, amphiphilic amino acid residues; '**T**' codes for hydrophobic residues. Fixing the composition of the 2nd position in the codon reduces the total number of high resolution dopes searched from 287^3 to 287^2 . Background gray bins indicate that much of H-M space was inaccessible. Maximum group probabilities for these three subsets of amino acids are given in Table 3.1 at synthesizer resolution.

Table 3.1. Numerical parameters for 19 different "synthesizer resolution" doping schemes. Columns are grouped according to the targeted amino acid subsets (in single letter code). Each group of columns has two headings: one describing a general classification of the target set and a specification of the target set itself. The first four groups are labeled by the general class 'C2=X', where X={N,A,C,T}, indicating the constraint on the middle position. The fifth group contains an aromatic dope and the last six groups are labeled by the position in the example peptide described in the text. The 37 rows are grouped in five sections: 1) H, M, and P enumerate average hydrophathy, average molar volume, and normalized group probability of the amino acid subset (p_G is normalized by the theoretical maximum of n^{-n} , where n is the number of acids in the subset); 2) A1, C1, G1, T1 are the fractional (dope) probabilities for these four nucleotides in the first codon position; 3) second position dopes; 4) third position dopes; 5) fractional probabilities of the resulting amino acid residues (X represents stop codons).

	C2=N	C2=A	C2=C	C2=T	Aromatic	N-7	N-4	N-3	N+3	N+4	N+7								
	all	DEHKNQY	SPAT	FILMV	FWY	GFILMSTVW	AGV	VAIL	SAV	FILMWAY	KNRST								
H	-0.31	-3.20	-3.20	-0.33	3.57	3.57	1.36	0.97	0.97	1.87	2.00	2.13	2.00	2.12	2.25	0.08	0.08	-2.35	-2.35
M	136.28	146.64	146.64	104.10	161.03	161.03	172.31	126.25	126.25	96.3	133.47	127.85	126.87	123.97	121.07	130.52	130.52	130.15	130.15
P	0.07	0.39	0.39	1.00	0.80	0.80	0.15	0.05	0.05	1.0	0.20	0.20	0.42	0.42	0.42	3.6e-4	3.6e-4	0.80	0.80
A1	0.25	0.25	0.25	0.25	0.33	0.33	0.00	0.33	0.33	0.00	0.33	0.33	0.00	0.00	0.00	0.33	0.33	1.00	1.00
C1	0.25	0.25	0.25	0.25	0.00	0.00	0.00	0.00	0.00	0.00	0.33	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
G1	0.25	0.25	0.25	0.25	0.33	0.33	0.00	0.33	0.33	1.00	0.33	0.33	0.50	0.50	0.50	0.33	0.33	0.00	0.00
T1	0.25	0.25	0.25	0.25	0.33	0.33	1.00	0.33	0.33	0.00	0.00	0.33	0.50	0.50	0.50	0.33	0.33	0.00	0.00
A2	0.25	1.00	1.00	0.00	0.00	0.00	0.33	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.25	0.25	0.33	0.33
C2	0.25	0.00	0.00	1.00	0.00	0.00	0.00	0.33	0.33	0.33	0.50	0.50	0.50	0.50	0.50	0.25	0.25	0.33	0.33
G2	0.25	0.00	0.00	0.00	0.00	0.00	0.33	0.33	0.33	0.33	0.00	0.00	0.00	0.00	0.00	0.25	0.25	0.33	0.33
T2	0.25	0.00	0.00	0.00	1.00	1.00	0.33	0.33	0.33	0.33	0.50	0.50	0.50	0.50	0.50	0.25	0.25	0.00	0.00
A3	0.00	0.50	0.00	1.00	0.00	0.00	0.00	0.00	0.00	1.00	1.00	1.00	0.00	0.50	1.00	0.00	0.00	0.50	0.00
C3	0.50	0.50	0.50	0.00	0.00	0.50	0.33	0.00	0.50	0.00	0.00	0.00	1.00	0.50	0.00	0.00	0.50	0.50	0.50
G3	0.50	0.00	0.50	0.00	0.50	0.50	0.33	0.50	0.50	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.50	0.00	0.50
T3	0.00	0.00	0.00	0.00	0.50	0.00	0.33	0.50	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.50	0.00	0.00	0.00
A	0.062	0.000	0.000	0.250	0.000	0.000	0.000	0.111	0.111	0.333	0.167	0.167	0.250	0.250	0.250	0.083	0.083	0.000	0.000
C	0.031	0.000	0.000	0.000	0.000	0.000	0.222	0.056	0.056	0.000	0.000	0.000	0.000	0.000	0.000	0.042	0.042	0.000	0.000
D	0.031	0.125	0.125	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.042	0.042	0.000	0.000
E	0.031	0.125	0.125	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.042	0.042	0.000	0.000
F	0.031	0.000	0.000	0.000	0.167	0.167	0.222	0.056	0.056	0.000	0.000	0.000	0.250	0.125	0.000	0.042	0.042	0.000	0.000
G	0.062	0.000	0.000	0.000	0.000	0.000	0.000	0.111	0.111	0.333	0.000	0.000	0.000	0.000	0.000	0.083	0.083	0.000	0.000
H	0.031	0.125	0.125	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
I	0.031	0.000	0.000	0.000	0.167	0.167	0.000	0.056	0.056	0.000	0.167	0.167	0.000	0.000	0.000	0.042	0.042	0.000	0.000
K	0.031	0.125	0.125	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.042	0.042	0.167	0.167
L	0.094	0.000	0.000	0.000	0.167	0.167	0.111	0.056	0.056	0.000	0.167	0.167	0.000	0.125	0.250	0.042	0.042	0.000	0.000
M	0.031	0.000	0.000	0.000	0.167	0.167	0.000	0.056	0.056	0.000	0.000	0.000	0.000	0.000	0.000	0.042	0.042	0.000	0.000
N	0.031	0.125	0.125	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.042	0.042	0.167	0.167
P	0.062	0.000	0.000	0.250	0.000	0.000	0.000	0.000	0.000	0.000	0.167	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Q	0.031	0.125	0.125	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
R	0.094	0.000	0.000	0.000	0.000	0.000	0.000	0.056	0.056	0.000	0.000	0.000	0.000	0.000	0.000	0.042	0.042	0.167	0.167
S	0.094	0.000	0.000	0.250	0.000	0.000	0.000	0.167	0.167	0.000	0.000	0.167	0.250	0.250	0.250	0.125	0.125	0.167	0.167
T	0.062	0.000	0.000	0.250	0.000	0.000	0.000	0.111	0.111	0.000	0.167	0.167	0.000	0.000	0.000	0.083	0.083	0.333	0.333
V	0.062	0.000	0.000	0.000	0.333	0.333	0.000	0.111	0.111	0.333	0.167	0.167	0.250	0.250	0.250	0.083	0.083	0.000	0.000
W	0.031	0.000	0.000	0.000	0.000	0.000	0.111	0.056	0.056	0.000	0.000	0.000	0.000	0.000	0.000	0.042	0.042	0.000	0.000
Y	0.031	0.125	0.125	0.000	0.000	0.000	0.222	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.042	0.042	0.000	0.000
X	0.031	0.125	0.125	0.000	0.000	0.000	0.111	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.042	0.042	0.000	0.000

As a practical example of the utility of this method, consider the design of a combinatorial cassette for a bacteriochlorophyll (bch) binding site in the β -subunit of the light harvesting II (LHII) polypeptide from *Rhodobacter capsulatus*. The bch binding site is modeled as a histidine residue within a transmembrane alpha helix²⁴. In other LH antennae, mutagenesis data indicate that this histidine residue is essential for the coordinate covalent bond to the Mg of the bch and that the residue at position N-4 must be small and at least moderately hydrophobic²⁵. The wild-type *Rhodobacter capsulatus* sequence is: G A M A L V A H I L S A I A I. In this example, we consider mutagenesis along the histidine (N = 0) side of the helix, at positions: N-7 = Gly, N-4 = Ala, N-3 = Leu, N+3 = Ser, N+4 = Ala, and N+7 = Thr. Phylogenetic data²⁴ from 29 known sequences of the β subunits reveals other functional residues at these sites (single letter code): N-7 = G,F,I,L,M,S,T,V,W; N-4 = A,G,V; N-3 = V,A,I,L; N+3 = S,A,V; N+4 = F,I,L,M,W,A,Y; N+7 = K,N,R,S,T. The central histidine is conserved in all 29 sequences.

For all six sites showing phylogenetic sequence variation, a doping is engineered which maximizes the group probability of the known substitutions while minimizing all other amino acids. Since equiprobable **A**, **C**, **G**, and **T** in all three positions of the codon yields all 20 amino acids, we are guaranteed a solution for any subset of amino acids. Figure 3.2 and Table 3.1 display and enumerate the best dopes for these six subsets of amino acid residues at high resolution and synthesizer resolution, respectively. These solutions can be rationalized by inspection of Figure 3.1.

Nucleotide dopings yielding the same group probability are referred to as degenerate solutions. Several examples are shown in Table 3.1. Breaking such degeneracies might involve expert rules,

such as accepting only **G** and **C** in the third position of the codon for **G-C** rich organisms. As the resolution of doping increases (given constant size of H-M bins) degeneracies become more frequent. In the program that generated Figure 3.2, only the most recently calculated high value of p_G is stored. This has no effect on the display.

Full implementation of the high resolution dopes used in Figure 3.2 must await the construction of automated DNA synthesizers capable of doping at an accuracy of at least 0.1 (i.e. 10%). Over 23 million different dopes are possible at this resolution as compared to only 3375 on current synthesizers. These two numbers should be compared to the $2^{20}-1$ possible subsets of amino acids (1,048,575). It should be noted that the dopes which can be implemented on current synthesizers are in general sufficient to provide better dopes than **NN(G,C)** for most of these subsets.

3.2.3 Acknowledgement

This work was supported by NIH GM42645, DOE DE-FG02-90ER20019 and the Human Frontiers Science Program. A.P.A. is supported by an NIH Biophysics Training Grant. We thank Mary M. Yang, Edward J. Bylina and George D. Rose for a critical reading of this manuscript.

3.3 Epilogue

The phylogenetic dopes defined in the above paper have been used by Ellen Goldman in the Youvan laboratory. The very preliminary results indicate that use of such a dope significantly increases throughput of functional mutants. This is useful for applications in which one is fine tuning the function of a given protein. However, phylogenies are not always available and may not be the best way of

choosing the target amino acids. Further, the choice of which sites to mutagenize is also extremely difficult and relies on expert knowledge. The choice of target sites may be facilitated by using molecular dynamics and energy minimization packages, crystal structures, or sequence analyses like those of Garnier²⁶, Chou and Fasman²⁷⁻²⁹, and Eisenberg^{30, 31}, and by compiling previous mutagenesis data. The choice of which amino acids to use in the dope may be made by, for example, grouping amino acids by physical class, phylogeny or evolutionary distance, or by the form of feedback described in Chapter 4. In each case an assumption is made that the restriction of the amino acid set discards more uninteresting proteins than desired ones.

The last topic to discuss here is the choice of the optimization function p_G . This function was chosen to maximize the probability of observing *all* the amino acids in the target set at as close to equal probability as possible. No amino acid in the set is allowed to be discarded. This property of equation 3.5 may be proven in a manner analogous to the method used to prove that equiprobability of appearance of the alphabet characters maximizes Shannon's entropy.¹⁷ In this proof we assume that the probabilities of the targeted amino acids sum to one, i.e. that the dope produces only acids in the target set (see, for example, the SPAT dope). This assumption may be relaxed by noticing that however probability is distributed within the target set, this portion of the total probability may be renormalized to one. To facilitate the proof we take the logarithm of Eq. 3.5 noting that this will not change the position of the maximum of the function, only its value. Equation 3.5 then becomes:

$$\log(p_G(p_1, p_2, \dots, p_n)) = \sum_{i=1}^n \log(p_i) \quad \text{Eq. 3.6}$$

where we have used the property that $\log(AB) = \log(A) + \log(B)$. Next we use an inequality which is true for any continuous convex function ϕ

$$\phi\left(\frac{1}{n} \sum_{i=1}^n a_i\right) \leq \frac{1}{n} \sum_{i=1}^n \phi(a_i) \quad \text{Eq. 3.7}$$

where the a_i are positive numbers, and n is the number of a_i in question. The simple geometric interpretation of this equation is that a function $\phi(x)$ must always lie below the linear function defined in the range spanned by the a_i . We assign $a_i = p_i$ and $\phi(x) = -\log(x)$ and remembering that the p_i sum to one we have:

$$\phi\left(\frac{1}{n}\right) = -\log\left(\frac{1}{n}\right) = -\frac{1}{n} \log\left(p_G\left(\frac{1}{n}, \frac{1}{n}, \frac{1}{n}, \dots, \frac{1}{n}\right)\right) \leq \frac{1}{n} \sum_{i=1}^n \phi(p_i) = -\frac{1}{n} \sum_{i=1}^n \log(p_i) = -\frac{1}{n} \log(p_G(p_1, p_2, \dots, p_n)) \quad \text{Eq. 3.8}$$

from which we find:

$$\log\left(p_G\left(\frac{1}{n}, \frac{1}{n}, \frac{1}{n}, \dots, \frac{1}{n}\right)\right) \geq \log(p_G(p_1, p_2, \dots, p_n)) \quad \text{Eq. 3.9}$$

Thus, the maximum value of our function is obtained when all the subset probabilities are the same and sum to one. Another proof is easily obtained using a LaGrange parameter for the constraint that the probabilities must sum to one. In this case one once again uses the logarithm of the p_G and simply sets the derivative equal to zero.

The property of p_G to be maximum for equiprobability and to conserve all amino acids in the target set may be undesirable in some cases. For example, in Chapter 6 it is demonstrated that using the SSD function may be optimal when Recursive Ensemble Mutagenesis (Chapter 4) is made consistent with DNA synthesizer resolution dopes. SSD better preserves the observed amino acid distribution and thus, hypothetically, will reproduce amino acids sequences more consistent with the data used to generate the choice of target amino acid set for each site. However, amino acids may be lost from the target set to achieve a minimum SSD.

The next Chapter defines a way of automatically choosing the target set by starting from a small random sampling of sequence space and using the information gained to create a positive feedback loop to refine dopes until a library encoding a large set of proteins fitting a desired criterion are produced. In Chapter 6, the work from this chapter will be merged with the method described in Chapter 4 to define a practical experimental protocol for Recursive Ensemble Mutagenesis.

7. Balle, A., J. H. Drenth, H. B. Jensen and T. L. Blundell. 1980. Local Complement of Protein Sequences and Structures to Protein Models and Design. *JIBS* 15: 238-240.

8. Malhotra, M. D. and H. L. Heymaker. 1985. Tertiary Structure Determination of the U₂ of Ambiguously Synthesized Oligonucleotides by Subsequent Synthesis Immediately 5' of an ATG Initiation Codon. *Nucl. Acids Res.* 13: 3112-3121.

9. Liu, W. A. and H. T. Saenger. 1988. Alternative Packing Arrangements in the Hydrophobic Core of λ repressor. *Nature* 339: 31-35.

3.4 References

1. Yang, M. M. and D. C. Youvan. 1988. Applications of Imaging Spectroscopy in Molecular Biology: I. Screening Photosynthetic Bacteria. *Bio/Technology* **6**: 939-942.
2. Arkin, A. P. and D. C. Youvan. 1992. Optimizing Nucleotide Mixtures to Encode Specific Subsets of Amino Acids for Semi-Random Mutagenesis. *Bio/Technology* **10**: 297-300.
3. Hermes, J. D., S. C. Blacklow and J. R. Knowles. 1990. Searching Sequence Space by Definable Random Mutagenesis- Improving the Catalytic Potency of an Enzyme. *Proc. Natl. Acad. Sci. USA* **87**: 696-700.
4. Roberts, B. L., W. Markland, A. C. Ley, R. B. Kent, D. W. White, S. K. Guterman and R. C. Ladner. 1992. Directed Evolution of a Protein: Selection of Potent Neutrophil Elastase Inhibitors Displayed on M13 Fusion Phage. *Proc. Natl. Acad. Sci. USA* **89**: 2429-2433.
5. Robles, S. J. 1990. Partial Symmetrization of the Photosynthetic Reaction Center. *Science* **248**: 1402-1405.
6. Tuerk, C. and L. Gold. 1990. Systematic Evolution of Ligands by Exponential Enrichment: RNA Ligands to Bacteriophage T4 DNA Polymerase. *Science* **249**: 505-510.
7. Sali, A., J. P. Overington, M. S. Johnson and T. L. Blundell. 1990. From Comparisons of Protein Sequences and Structures to Protein Modelling and Design. *TIBS* **15**: 235-240.
8. Matteucci, M. D. and H. L. Heyneker. 1983. Targeted Random Mutagenesis: The Use of Ambiguously Synthesized Oligonucleotides to Mutagenize Sequences Immediately 5' of an ATG Initiation Codon. *Nucl. Acids Res.* **11**: 3113-3121.
9. Lim, W. A. and R. T. Sauer. 1989. Alternative Packing Arrangements in the Hydrophobic Core of λ repressor. *Nature* **339**: 31-36.

10. Hermes, J. D., S. M. Parekh, S. C. Blacklow, H. Koster and J. R. Knowles. 1989. A Reliable Method for Random Mutagenesis: The Generation of Mutant Libraries Using Spiked Oligo-Deoxyribonucleotide Primers. *Gene* **84**: 143-151.
11. Oliphant, A. R., A. L. Nussbaum and K. Struhl. 1986. Cloning of Random-Sequence Oligodeoxynucleotides. *Gene* **44**: 177-183.
12. Reidhaar-Olson, J. F. and R. T. Sauer. 1988. Combinatorial Cassette Mutagenesis as a Probe of the Informational Content of Protein Sequences. *Science* **241**: 53-57.
13. Reidhaar-Olson, J. F., J. U. Bowie, R. M. Breyer, J. C. Hu, K. L. Knight, W. A. Lim, M. C. Mossing, D. A. Parsell, K. R. Shoemaker and R. T. Sauer. 1991. Random Mutagenesis of Protein Sequences Using Oligonucleotide Cassettes. *Methods in Enzymology* **208**: 564-587.
14. Wang, X. and G. J. Pielak. 1991. Temperature-Sensitive Variants of *Saccharomyces cerevisiae* Iso-1-cytochrome *c* Produced by Random Mutagenesis of Codons 43-54. *J. Mol. Biol.* **221**: 97-105.
15. Kyte, J. and R. F. Doolittle. 1982. A Simple Method for Display the Hydrophobic Character of a Protein. *J. Mol. Biol.* **157**: 105-132.
16. Zamyatnin, A. A. 1972. Protein Volume in Solution. *Prog. Biophys. Mol. Biol.* **157**: 105-132.
17. Khinchin, A. I. 1957. *Mathematical Foundations of Information Theory*. Dover Publications, Inc., New York.
18. Lim, W. A. and R. T. Sauer. 1991. The Role of Internal Packing Interactions in Determining the Structure and Stability of a Protein. *J. Mol. Biol.* **219**: 359-376.
19. Chan, S. H. and K. A. Dill. 1990. Origins of Structure in Globular Proteins. *PNAS* **87**: 6388-6392.
20. Lee, C. and M. Levitt. 1991. Accurate Prediction of the Stability and Activity Effects of Site-Directed Mutagenesis on a Protein Core. *Nature* **352**: 448-451.

21. Sjostrom, M. and S. Wold. 1985. A Multivariate Study of the Relationship Between the Genetic Code and the Physical-Chemical Properties of Amino Acids. *J. Mol. Evol.* **22**: 272-277.
22. Yang, M. M., W. J. Coleman and D. C. Youvan. 1990. Genetic Coding Algorithms For Engineering Membrane Proteins. *In* Reaction Centers of Photosynthetic Bacteria. M.-E. Michel-Beyerle. (Ed.) (Springer-Verlag, Germany) 209-218.
23. Youvan, D. C. 1991. Photosynthetic Reaction Centers: Interfacing Molecular Genetics and Optical Spectroscopy. *TIBS* **16**: 145-149.
24. Zuber, H. 1990. Consideration on the Structural Principles of the Antenna Complexes of Phototrophic bacteria. *In* Molecular Biology of Membrane-Bound Complexes in Phototrophic Bacteria. G. Drews and E. A. Dawes. (Ed.) (Plenum Press, New York) 161-180.
25. Bylina, E. J., S. J. Robles and D. C. Youvan. 1988. Directed Mutation Affecting the Putative Bacteriochlorophyll-Binding Sites in the Light-Harvesting I of *Rhodobacter capsulatus*. *Isr. J. Chem.* **28**: 73-78.
26. Garnier, J. 1990. *In* Prediction of Protein Structure and the Principles of Protein Conformation. G. D. Fasman. (Ed.) (Plenum Press, New York)
27. Arkin, A. P., E. Goldman, S. J. Robles, W. J. Coleman, C. A. Goddard, M. M. Yang and D. C. Youvan. 1990. Applications of Imaging Spectroscopy in Molecular Biology II: Colony Screening Based on Absorption Spectra. *Bio/Technology* **8**: 746-749.
28. Fasman, G. D. 1989. Protein Conformational Prediction. *TIBS* **14**: 295-299.
29. Prevelige, P. and G. D. Fasman. 1990. Chou-Fasman Prediction of the Secondary Structure of Proteins: The Chou-Fasman-Prevelige Algorithm. *In* Prediction of Protein Structure and the Principles of Protein Conformation. G. D. Fasman. (Ed.) (Plenum Press, New York) 391-416.
30. Bowie, J. U., R. Luthy and D. Eisenberg. 1991. A Method to Identify Protein Sequences That Fold into a Known Three-Dimensional Structure. *Science* **253**: 164-170.

31. Luthy, R., J. U. Bowie and D. Eisenberg. 1992. Assessment of Protein Models with Three-Dimensional Profiles. *Nature* **356**: 83-85.

Recursive Ensemble Metagenesis

32. Hardy, Littlewood and Polya. 1934. *Inequalities*. Cambridge University Press, Cambridge, UK.

The following **Recursive Ensemble Mutagenesis** is published in the Proceedings of the National Academy of Sciences.

4.2 Simulations of Recursive Ensemble Mutagenesis

4.2.1 Tutorial

In this tutorial, we will explore the concept of Recursive Ensemble Mutagenesis (REM) as a method for developing a diverse set of models for a given task. The idea is to start with a single model and iteratively improve it by adding new models that are trained on the data that the current model gets wrong. This process is repeated until the model's performance is satisfactory. The key to REM is the use of a diverse set of models, which are trained on different subsets of the data. This diversity is achieved by using a technique called "bagging" (bootstrap aggregating), which involves sampling the data with replacement to create multiple training sets. Each training set is used to train a model, and the outputs of all models are combined to produce the final prediction. The tutorial will walk through the steps of implementing REM, from data preparation to model training and evaluation. It will also discuss the advantages of REM, such as its ability to reduce variance and improve model performance, and its application in various fields, including machine learning and bioinformatics.

4.2.2 Tutorial: Engineering the Protein Folding Problem

One major challenge in the solution to the protein folding problem is to accurately predict the 3-dimensional structure and function of proteins from primary amino acid sequence data. Although some progress has been made in predicting the secondary structure of proteins^{1,2}, no algorithm

4.1 Forward

The following paper has been communicated to the Proceeding of the National Academy of Sciences.

4.2 Simulations of Recursive Ensemble Mutagenesis

4.2.1 Summary

A new algorithm for protein engineering, termed Recursive Ensemble Mutagenesis (REM), has been developed to produce diverse populations of phenotypically related mutants whose members differ in amino acid sequence. This method uses a feedback mechanism to control successive rounds of combinatorial cassette mutagenesis. Starting from partially randomized "wild-type" DNA sequences, a highly parallel search of sequence space for peptides fitting an experimenter's criteria is performed. Each iteration uses information gained from the previous rounds to search the space more efficiently. Simulations of the technique indicate that, under a variety of conditions, the algorithm can rapidly produce a diverse population of proteins fitting specific criteria. In the experimental analog, genetic selection or screening applied during REM should force the evolution of an ensemble of mutants to a targeted cluster of related phenotypes.

4.2.2 Reverse Engineering the Protein Folding Problem

One might envision the solution to the protein folding problem as a complex algorithm that accurately predicts the 3-dimensional structure and function of proteins from primary amino acid sequence data. Although some progress has been made in predicting the secondary structure of proteins¹⁻⁹, no algorithm

exists that can decode an amino acid sequence into a 3D structure and predict the chemical properties of the resultant protein. If such an algorithm did exist, one could design and engineer proteins to solve problems in fields as diverse as industrial catalysis¹⁰, bioremediation¹¹ and medicine.¹² Though much progress has been made towards a semi-empirical solution to the protein folding problem, a complete solution to this quandary seems to lie in the distant future.⁶

From the viewpoint of a molecular geneticist, and in the absence of a solution to the protein folding problem, how does one effectively search for new proteins with desired structures and functions? In this communication, we present computer simulations of an algorithm that efficiently searches "sequence space"¹³ for proteins with specified properties, while treating the protein folding problem as a black box. Each step in this simulated process is exactly analogous to standard laboratory processes: DNA synthesis, cloning, expression, screening, and sequencing. Both the simulated process and the putative experimental process are termed "recursive" because information gained in one round of mutagenesis is used to control the next. We have been successful in simulating recursive ensemble mutagenesis (REM) on as many as eight interactive amino acid sites and have embarked upon analogous experiments with model proteins.

Simultaneously randomizing eight amino acid positions in a protein leads to a sequence complexity of 20^8 , or over 25 billion different sequences. The principle advantage of REM over other mutagenesis methods is that one can assay a relative small volume (e.g. 10,000 mutants) in this large sequence space, find a few "positives", and generate a much larger fraction of acceptable sequences in the next round of mutagenesis. As such, REM is based on

repeated use of combinatorial cassette mutagenesis¹⁴⁻¹⁷ (CCM) directed by an algorithm that actively adjusts nucleotide compositions for DNA synthesis.

While most of this communication focuses on complex simulations of parameters that might affect the technique, the experimental flowchart for REM is relatively simple. First, several (e.g. eight) positions within a protein are selected for random mutagenesis, possibly by the criterion of proximity to an "active" site thought to be of mechanistic importance. These positions need not be contiguous. The zeroth iteration of REM begins by simultaneously replacing all eight codons with **NNN** (i.e. equiprobable **A,C,G,T** in all three positions of the mutagenized codons). Second, the CCM library is expressed and screened according to specific phenotypic criteria. It is important that at least a few "positives" are found in the zeroth round of REM. This depends on how many different amino acids are acceptable per site, the number of sequences (i.e. colonies) screened, and the stringency of the screening or selection criteria. In the third experimental step of REM, DNA from the positive colonies is extracted and sequenced. Clearly, some of the sites may tolerate only a few amino acid substitutions, making it possible to restrict the "doping" of codons in the next cycle of mutagenesis to something less complex than **NNN**¹⁸⁻²¹. For example, a site that requires hydrophobic residues could be redoped with **NTN**.^{18, 19, 21} The mechanisms by which REM successfully adjusts nucleotide "dopes" is the subject of much of this communication.

Our simulations show that REM is very robust in finding sequences that fit a variety of selection algorithms. In some cases, we have attempted to make such algorithms physically realistic, incorporating data from wild-type and physicochemical parameters

known to be of importance in protein structure and function: hydrophathy²², molar volume²³, and parameters related to propensities for forming various types of structures.¹ REM simulations using physically irrelevant (e.g. lexicographic) rules also demonstrate convergence (albeit weaker) through adaptive feedback. This strongly suggests that the method is experimentally viable regardless of the form of the solution to the protein folding problem.

In both simulated and experimental REM, it is very important to count only "unique" mutants, those with different combinations of amino acid residues at the mutagenized sites. This avoids the problem of the entire ensemble of mutants becoming identical, which is termed a clonal jackpot by molecular geneticists. As the size and diversity of the ensemble of mutants increases, it becomes more probable that the experimenter will find a mutant with exceptional properties. Thus one circumvents the problems associated with the *de novo* design of proteins and effectively reverse engineers the protein folding problem.

In hindsight, we have recognized that the mathematical format of REM resembles techniques developed in the field of Genetic Algorithms²⁴⁻²⁶ (GAs). To date, and despite their name, GAs have not been applied to problems in genetics.²⁵ In the discussion, we shall elaborate on the non-trivial differences between the mathematical basis of REM and GAs, and give reasons why the former converge more rapidly in this domain.

4.2.3 Description and Results of Simulations

4.2.3.1 Theory and Description of Algorithm

We define *sequence space* as the set of all possible protein sequences that can arise through random mutagenesis of N amino acid residues. For purposes of REM, four subsets exist: 1) The *wild-type subset*, a set of cardinality one which contains the sequence of the protein being mutagenized. 2) The *selection subset*, which contains all protein sequences fitting selection or screening criteria. 3) The *pseudo-wild-type subset*, which phenotypically resembles wild-type to some specified degree. 4) The *null subset*, containing all remaining protein sequences, including those which do not fold or express. In special cases, the pseudo-wild-type subset can become the selection subset, yielding diverse sequences which are iso-functional with wild-type.

The genetic engineer (in this simulation) knows only the wild-type sequence and has no way of knowing the sequences within the selection subset. Based on expert opinion, the engineer first reduces the search volume by choosing a relatively small number of amino acids to mutagenize (commensurate with the fraction of the population to be screened). Typically, this might include all of the residues in an active site. However, this reduced volume (20^N for N sites) is still too large to search thoroughly if N is greater than about six amino acid residues. As will be shown, REM efficiently finds a path from the wild-type set to the selection subset.

A mechanical analogy for the algorithm is shown in Figure 4.1. The input cylinder at the left of the figure is partitioned into S columns each containing a DNA sequence. Each sequence represents the $3N$ nucleotides needed to mutagenize N amino acids in the wild-

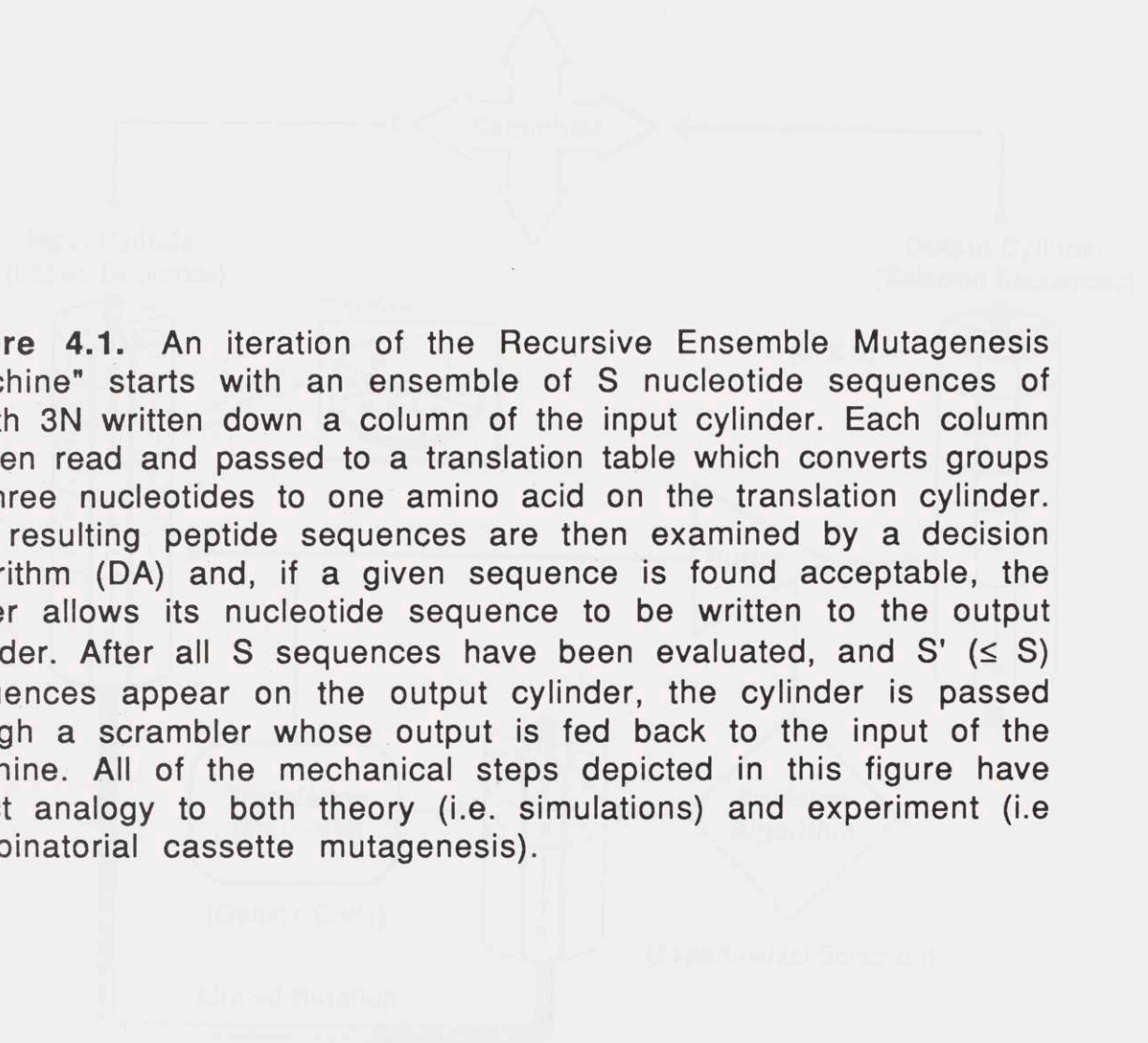
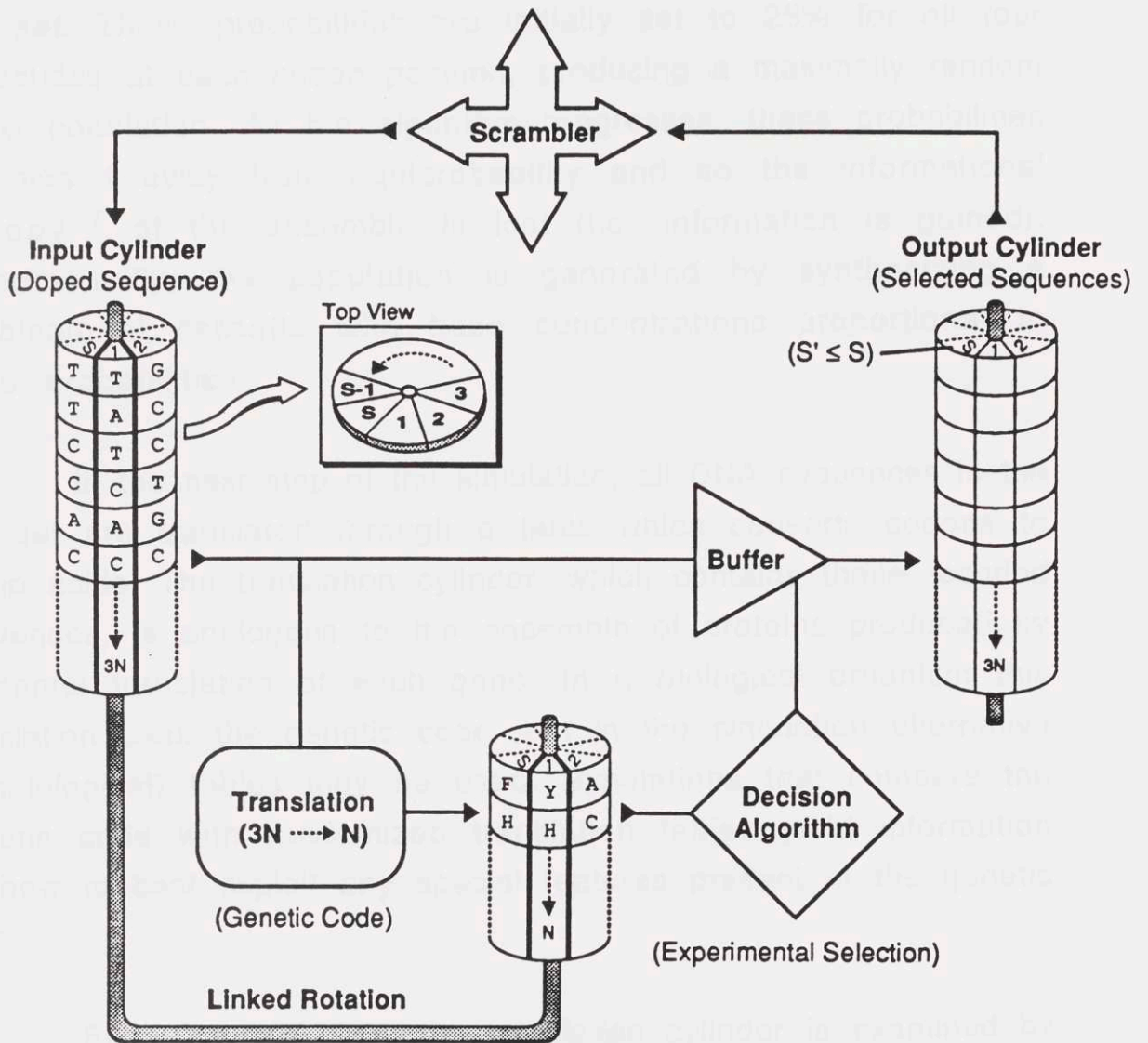


Figure 4.1. An iteration of the Recursive Ensemble Mutagenesis "machine" starts with an ensemble of S nucleotide sequences of length $3N$ written down a column of the input cylinder. Each column is then read and passed to a translation table which converts groups of three nucleotides to one amino acid on the translation cylinder. The resulting peptide sequences are then examined by a decision algorithm (DA) and, if a given sequence is found acceptable, the buffer allows its nucleotide sequence to be written to the output cylinder. After all S sequences have been evaluated, and S' ($\leq S$) sequences appear on the output cylinder, the cylinder is passed through a scrambler whose output is fed back to the input of the machine. All of the mechanical steps depicted in this figure have exact analogy to both theory (i.e. simulations) and experiment (i.e. combinatorial cassette mutagenesis).



type protein. These sites are not necessarily contiguous. The population is generated by assigning probabilities for nucleotides at every one of the $3N$ positions and then randomly creating S sequences based on this distribution. This population is called the *test set*. These probabilities are initially set to 25% for all four nucleotides at each codon position, producing a maximally random (DNA) population. As the algorithm progresses, these probabilities are biased away from equiprobability and so the informational entropy²⁷ of the ensemble is lost (i.e. information is gained). Experimentally, the population is generated by synthesizing a combinatorial cassette with base concentrations proportional to these probabilities.

In the next step of the simulation, all DNA sequences in the test set are translated through a table which converts codons to amino acids. The translation cylinder, which contains these recoded sequences, is analogous to the ensemble of proteins produced by ribosomal translation of each gene. In a biological organism this translation uses the genetic code, but in the simulation alternative (nonbiological) tables may be used. Simulations that compare the genetic code with randomized translation tables yield information on how to best exploit any special features present in the genetic code.

Each "protein" from the translation cylinder is examined by a *decision algorithm* (DA) and is either accepted or rejected based on a set of rules we term a *protein grammar*. This grammar can be a combination of heuristic rules for protein folding^{4, 7}, energy minimization^{2, 5}, or whatever specific criteria the engineer wishes to apply. [For purposes of this discussion, whether or not the protein grammar is biophysically correct is irrelevant]. If the peptide is accepted by the DA (equivalent to experimental "positives"), then the

gene is passed through the buffer and is written to the output cylinder (see Fig. 4.1). Otherwise, the gene is discarded. Once all S peptide sequences have been examined, the output cylinder will contain S' ($\leq S$) DNA sequences (*acceptance set*) and the cylinder is passed to a "scrambler". The number of different peptide sequences in S' is designated UM, for unique mutants. Experimentally, UM is an important parameter for the geneticist to maximize relative to the total number of colonies (i.e. sequences) screened.

To start a new REM iteration, the scrambler routine calculates the percentages of each nucleotide at every position in the output cylinder and then uses these percentages as probabilities to generate a new input cylinder. Since S' can be a relatively small number (depending on S and the *stringency* of the DA), the final distribution of bases per position on the new input cylinder may be slightly different than the output cylinder. Note that this method of generating the new test set is very different than in genetic algorithms which typically recombine "chromosomes" by pairwise crossings (see Discussion). The scrambler is an analogy for two experimental steps: 1) combining nucleotide data from the mutagenized sites by either batch^{18, 28} or standard DNA sequencing, and 2) constructing a new library of combinatorial cassette mutants through DNA synthesis and cloning.

4.2.3.2 General Aspects of REM Simulations

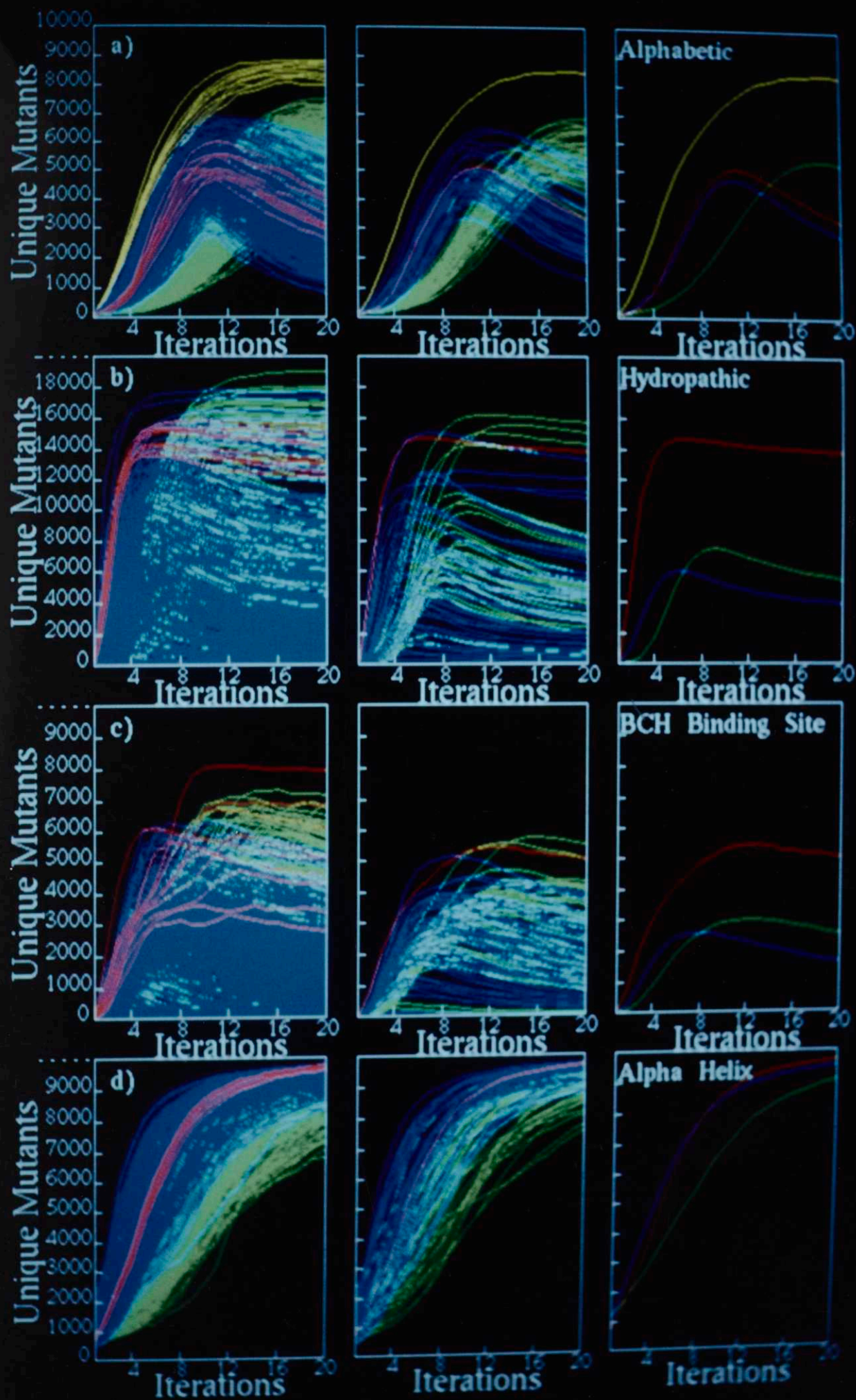
The DAs used in the REM simulations are not designed to mimic the biophysics of protein structure and function in a way that requires a solution to the protein folding problem. REM simulations are important because insight can be gained into parameters that are important for efficient feedback and amplification of the number of unique mutants falling within a selection subset. For example, if

REM decision algorithms involve constraints on amino acid hydrophathy, then we expect strong biasing in the second position of the codon.^{18, 20} In fact, since the REM algorithm is meant to function under any DA, physically irrelevant rules that are very simple (e.g. requiring amino acids to be in alphabetic order within a octapeptide) can be employed to test REM behavior under a variety of experimentally relevant parameters (such as sampling size).

Figure 4.2 shows plots of UM vs. iterations of REM for four different DAs. Each simulation uses at least three different classes of translation tables. The most general class is termed a *random code* class, wherein codons are randomly selected without replacement from the 64 possible triplets. The first 21 selections are assigned to the twenty amino-acids and a stop. Thereafter, the codons are assigned at random to the 21 possible elements of this set. The next class of translation tables contains *shuffle codes*, which keep the codon groupings intact (relative to the genetic code) but randomly reassign the amino acids and the stop command to each grouping. This maintains the codon degeneracies found in the genetic code but removes any other structuring (e.g. **NTN** codes for hydrophobic residues). The genetic code is more closely related to shuffle codes than random codes. An *alphabetic code** maintains the number of codons assigned to each amino acid but assignment is made in alphabetic-order. Thus, alanine (single letter code A) is assigned **AAA, AAC, AAG, AAT**, while tyrosine (Y) is assigned **TTG** and **TTT**.

* The alphabetic translation table is as follows: A: **AAA, AAC, AAG, AAT**; C: **ACA, ACC**; D: **ACG, ACT**; E: **AGA, AGC**; F: **AGG, AGT**; G: **ATA, ATC, ATG, ATT**; H: **CAA, CAC**; I: **CAG, CAT, CCA**; K: **CCC, CCG**; L: **CCT, CGA, CGC, CGG, CGT, CTA**; M: **CTC**; N: **CTG, CTT**; O (Stop): **GAA, GAC, GAG**; P: **GAT, GCA, GCC, GCG**; Q: **GCT, GGA**; R: **GGC, GGG, GGT, GTA, GTC, GTG**; S: **GTT, TAA, TAC, TAG, TAT, TCA**; T: **TCC, TCG, TCT, TGA**; V: **TGC, TGG, TGT, TTA**; W: **TTC**; Y: **TTG, TTT**.

Figure 4.2. REM simulations using four different decision algorithms. The plots show the number of unique peptide sequences (i.e. unique mutants, UM) accepted by the decision algorithm versus iteration number. Each simulation ran for twenty iterations on an octapeptide. There are three panels (left to right) for each simulation. The first panel shows seed plots for ten initial populations for all translation tables studied (see text). The second panel shows performance plots for each translation table. The third panel shows performance averages for various translation classes: random codes in green, shuffle codes in blue, the genetic code in red, and an alphabetic code in yellow. The four DAs used in a-d require: a) an alphabetic-ordering of the amino acids names (one letter code, no more than one out-of-order allowed), b) residues with Kyte and Doolittle hydropathy values less than zero at sites 1 and 4 and residues with hydropathy greater than zero everywhere else, c) a spectrally-shifted bacteriochlorophyll binding site constructed according to expert rules (see text), and d) a Chou-Fasman predicted alpha helix.



Each REM simulation displayed in Figure 4.2 was run with 45 different random codes (green), 45 different shuffle codes (blue), and the genetic code (red). In the upper set of panels, the alphabetic code (yellow) was also included. Simulations were run for twenty iterations with the test set size, S , equal to 10,000 or 20,000 (see labeling on the y-axis). In all cases the number of mutational sites was eight (i.e. octapeptide) and the initial nucleotide probabilities for all 24 nucleotide sites were equiprobable ($A,C,G,T = 25\%$). To determine the effects of the initial test set on the algorithm, ten different initial populations (*seeds*) were randomly generated.

In Figure 4.2, the three panels shown for each simulation represent different degrees of data averaging. The first panel shows plots of simulations of every translation table run with ten different seeds, yielding *seed plots*. The second panel shows *performance plots* which are an average of ten seed plots each run with the same translation table and DA. Performance plots can be used to rate the overall performance of the algorithm independent of the initial ensemble. Finally, the third panel displays *performance averages* which are averages of performance plots over a specific code class. Performance averages can be used to rate the overall performance of the algorithm as a function of the translation class: alphabetic, genetic, random, or shuffle. Performance plots and performance averages are identical for the alphabetic and the genetic code classes because there is only one example of each.

4.2.3.3 REM Simulations Using an "Alphabetic" Decision Algorithm

Figure 4.2a displays simulations using a DA that requires the single letter codes for the amino acid residues in an octapeptide to be in alphabetic-order. The DA allows one out-of-order residue

after which the order may start over again. Thus, the algorithm allows sequences like ACDFGHMQ and ADCDFGHM but not AACDFHGM or AEDAGHMQ. This simulation was run to highlight several important issues which bear on REM in general. First, notice that in all cases the packing of seed plots is fairly tight. This occurs because the numerous ways an alphabetic sequence may be formed results in very little epistasis between sub-motifs. *Epistasis*^{26, 29} is defined as mutual competition between sequence motifs. The alphabetic DA appears to require a single, almost non-exclusionary motif. Thus, the limitation on the entropy due to selection of one or another specific sub-motifs is lifted and maximum *amplification* (i.e. gain in UM as a function of iteration number) can be achieved. As expected, the alphabetic code (yellow curves) demonstrate the highest amplification of all translation classes. This may be attributed to the excellent match between the alphabetic DA grammar and the alphabetic translation table.

The alphabetic REM simulation enables us to explore a general feature of the algorithm pertaining to the match between the DA grammar and features within the translation table. The REM algorithm partially relies on the formation of new acceptable sequence patterns from the cross-products formed by the averaging action of the scrambler. A *cross-product* is defined as a new pattern created by the statistical recombination of two or more codons, e.g. cross-products of **CAC** and **CTG** are **CAG** and **CTC**. Cross-products create recombinant sequences in the next test set. Scrambling is more likely to produce new and acceptable motifs if codons with similar nucleotide compositions translate to amino acids with similar properties. Inherent structure in the translation table would affect the efficacy of the scrambling step for optimization. For the alphabetic code, recombinations which occur within a codon of two accepted sequences are likely to produce an alphabetically-close

codon and amino acid. Recombinations which also cause swaps of alphabetic runs are more likely to be acceptable because position in the gene is correlated to position in the alphabet.

The genetic code and shuffle codes for the alphabetic DA behave very differently than the random codes. Initially, the amplification is greater, but the UM quickly reaches a maximum and then decays rapidly. This implies that the way codons are assigned to the amino acids in the genetic code lends some initial advantage for amplification. For genetic and shuffle codes, the third codon position retains its "wobble" property, i.e. changing the identity of the third position base is unlikely to change the identity of the coded amino acid. This partial loss of a degree of freedom for the algorithm to employ in biasing the dopes results in a rapid determination of a consistent mutational scheme. The algorithm is forced to strongly bias the first two positions in the codon to fulfill the requirements of the DA. It is this rapid loss of complexity in combination with a strict rule which would cause a final over-biasing of the ensemble and a descent to a *clonal population* (i.e all sequences in the acceptance set are identical).

Finally, it is important to observe that REM simulations with random codes show significant amplification of the UM. However, the loss of intentional structure in these codes reduces the magnitude to which the DA can effectively bias the dopes. This means that sampling error is relied upon to select one type of motif over another. The properties of the selected motif will determine the efficiency of the amplification. These simulations show a greater spread in the seed plots, since information inherent in the initial test set becomes relatively important in determining the feedback pattern.

4.2.3.4 REM Simulations on an Hydropathically Constrained Domain

The REM simulations displayed in Figure 4.2b were performed with a simple hydrophathy constraint which requires hydrophilic residues (Kyte and Doolittle hydrophathy value less than zero) at octapeptide positions 1 and 5 and hydrophobic residues (values greater than zero) at the remaining six positions. A screening size of 10,000 was found to be too small for efficient functioning of the algorithm (often only one or two acceptable mutants were found), so S was increased to 20,000. It is noteworthy that the simulations using the genetic code perform differently than other translation classes with this DA.

For the hydrophatic DA, simulations using the genetic code amplify more rapidly than any other translation class. It is well known that the **A/T** disparity in the second position of a codon is predictive of the hydrophathy of the coded amino acid. This rather simplistic DA, requiring the selection of either hydrophobic or hydrophilic amino acids at specific positions, leads to a rapid biasing of the second codon positions of the codons (data not shown) and rapid amplification. Furthermore, if all eight codons were initially set to either **T** or **A** in the second position (hydrophobic and hydrophilic sites, respectively), the protein motifs produced from such an ensemble would very likely be acceptable by the DA, since only amino acids with similar hydrophathy values would be produced by the cross-product mechanism.

4.2.3.5 REM Simulation of a Complex Binding Site

Figure 4.2c shows simulations of a grammar devised to select for sequences that might create a spectroscopically perturbed bacteriochlorophyll (BCH) binding site.³⁰ Our interest in

this particular grammar stems from our technical capability to screen large numbers of colored colonies on petri dishes for spectroscopic shifts in the absorption spectra of light harvesting proteins. This grammar was constructed by examining phylogenetic data³¹ from numerous species and from mutagenesis data for *Rhodobacter capsulatus*.³² Based on these data, a complex DA was formulated: 1) The octapeptide must contain the motif (*a* x x x H), where *a* is a small amphiphilic residue (G, A, S or C) four sites to the left of the histidine (H) binding site. 2) One or two charged residues must be present in the octapeptide which may generate an electrochromic shift in the BCH near-infrared absorption spectrum.³³ 3) The average hydrophathy and molar volume of the octapeptide must be similar to the wild-type sequence (LAVLIHLL). The rejection threshold was based on the formula:

$$D = \sum_{i=1}^8 \{ [(H_m^i - H_w^i) / \Delta H]^2 + [(M_m^i - M_w^i) / \Delta M]^2 \} \quad \text{Eq. 4.1}$$

where *H* is the hydrophathy, and *M* is the molar volume. The subscripts *m,w* refer to mutant and wild-type sequences, and the superscript refers to the position in the peptide, respectively. ΔH and ΔM are the maximum hydrophathy and molar volume differences between pairs of amino acids, which is used for normalization. Sequences were rejected for *D* values greater than 2.0.

Since there are specific positional requirements for charged residues, hydrophobic residues, moderately sized residues, and a histidine, the selection set for the BCH DA is relatively small. This should lead to very strong biasing and epistasis. The seed plots for these simulations (4.2c) show greater spreads for UM (for a single translation table) than for the alphabetic (4.2a) or amphipathic (4.2b) rules. This is not surprising given the relative complexity of the rules used for selection. Both the location of the

binding site and the number and position of the charges are variable, therefore the initial population will almost certainly determine the final dominant sequence motifs. The minimal sequence differences in the ensembles resulting from one seed plot (data not shown), implies that during an actual experiment, it would be advantageous to run a number of populations through the mutagenesis protocol in parallel. Once again, simulations employing the genetic code are amplified more rapidly than other translation classes due to hydrophathy-molar volume constraints on the DA. However, all translation classes show significant amplification.

4.2.3.5 REM Simulations Involving an Alpha Helix

It is of interest to consider a physically realistic DA in which no correlation is expected with the genetic code. Such is the case for a DA which attempts to produce an ensemble of octapeptides that fold into a specific conformation. The Chou-Fasman DA uses modified code from the program described in reference 8 and their 64 protein database. Each amino acid residue is assigned a probability of being in an α -helix, a β -sheet, or a turn. To predict whether a given residue belongs to a given secondary structure, the propensity of the residue being in one of these three categories is averaged over four residues (tetrad propensity). A peptide is defined as "helical" and accepted by our DA if: 1) four or more consecutive helix tetrad values are greater than 100, 2) no residues in this helix nucleation site have higher tetrad values for sheet or turn, 3) no proline residues are found in the helix nucleation site, 4) no stop codons are encountered.

Figure 4.2d displays REM simulations using a Chou-Fasman prediction¹ for an alpha helix as the DA. Again, the seed curves are packed very tightly indicating a low sensitivity to initial conditions.

Simulations with all three classes of translation tables exhibit amplifications on the order of 22 times the initial population. This strongly suggests that REM leads to some amplification without correlation between the DA and the structure of the genetic code. Amplification is also minimized because the stringency of the Chou-Fasman DA is relatively low (compared to the other DAs we have investigated) and allows a large initial population through the selection.

4.2.3.6 Acceleration of REM

All the REM simulations shown in Figure 4.2 were run for 20 iterations. In many of the simulations, peak UM is reached between 10 and 20 iterations. Experimentally, each mutagenesis cycle will take from two weeks to a month, so the simulations are run for the experimental equivalent of two years. Obviously, a method for accelerating the rate at which the algorithm maximizes UM is desirable. An acceleration of the Chou-Fasman helix DA was simulated using an equation which extrapolates nucleotide dopes according to two parameters (A and s):

$$B_{syn}[i, j+1] = B_{gel}[i, j] + (B_{gel}[i, j] - B_{syn}[i, j]) \times A \times \text{EXP}\left(\frac{-(j+1)}{s}\right) \quad \text{Eq. 4.2}$$

This simulation was run with values of s ranging from 1 to 128 in powers of 2 and values of A ranging from 1 to 64. $B[i, j]$ is the fraction (0.0 - 1.0) of nucleotide B at position i and iteration j . The subscript 'syn' applies to synthesized DNA whereas the subscript 'gel' refers to the average nucleotide fractions read from the last set of sequencing gels on positive mutants. After the new nucleotide densities are calculated, the sum of **A, C, G, T** densities are normalized to one. The best parameters for this REM simulation

(Chou-Fasman alpha helix; same as Figure 2d) are: $A=4$ and $s=1$. Peak UM is achieved in only three iterations (data not shown).

Our accelerated REM simulations reveal that the best values for the pre-exponential and exponential factors in Eq. 4.2 correspond to a moderate initial acceleration which is sharply damped in later iterations. Damping prevents severe biasing which results in the formation of a clonal population. In this specific case, it is noteworthy that the peak UM under acceleration is greater than the UM for the unaccelerated population.

4.2.4 Conclusions: The Relation of REM to GAs, etc.

The application of artificial intelligence (AI) techniques in molecular genetics requires that researchers in both areas develop a mutually understandable language. We have attempted to present REM by using simulations directly related to experimental techniques that can be implemented by molecular geneticists conducting CCM. For further development of the algorithm itself, however, it is advantageous to also present REM in a form recognizable to the AI community.

Genetic Algorithms (GAs) are well known to the AI community as very powerful computational optimization techniques. GAs utilize *cross-over* and *mutational operators* to recombine *chromosomes* in a recursive manner such that the *fitness* of the population increases. An elementary GA might, for example, "evolve" a population of initially random 8 character capital letter strings (chromosomes) towards a target chromosome: "GENETICS". There are 26^8 possible strings of this length. With a 50% probability, one could find this specific chromosome by randomly generating and checking about 100 billion different strings. The power of GAs is

that one can reduce this search significantly by recombining and mutating chromosomes. For example, chromosomes can be ranked according to a "fitness" criterion (F), by counting the number of correct characters in the chromosome as a function of position. Chromosomes are allowed to mutate at a specified frequency, recombine pairwise by cross-over events, and be propagated into the next generation (by a variety of techniques) according to their fitness value. With a total population of 100 chromosomes, an efficient GA can converge on "GENETICS" in about 30 iterations, i.e. only 3000 strings are evaluated. GAs tend to preserve *schemata*. In this example, a single crossover between two chromosomes with $F=4$ (GENEWXYZ and ABCDTICS) in the penultimate iteration brings together the schemata "GENE" and "TICS". The mutational operator might have played an important step in a previous iteration by mutating the chromosome "PENEWXYZ" ($F=3$) to "GENEWXYZ". The explicit form of the fitness criterion need not be known for the GA to work. Molecular geneticists might draw an analogy between the GA "schemata" and a protein "structural motif", while chromosomes, mutations, and cross-overs convey similar concepts in both languages. Ironically, GAs have not been used in the field of genetics.

25

REM differs from elementary GAs in several important ways: 1) The REM decision algorithm evaluates one type of string (protein sequence) while optimizing another string (nucleotide sequence). The protein sequence (length = N) utilizes an alphabet of 20 characters, while the nucleotide sequence is 3N in length and uses 4 characters. In contrast, the GA characterized above regenerates and evaluates only one type of string: a chromosome constructed from the 26 characters of the alphabet. 2) REM generates the next test set by a stochastic process based on the probabilities of each character in the 3N string (i.e. probability of A,

C, G, and T at each position in the synthetic gene). In REM, all members of the population are scrambled with each other as opposed to the pairwise matings in GAs. 3) REM does not assign fitness values to sequences other than one (retain) or zero (discard). 4) Since no dependent probabilities are maintained between positions within a codon, the cross products that are generated in REM are dependent on the structure of the translation table (e.g. genetic code) as compared to the random mutagenesis found in GAs.

We have already shown that the rapid convergence of REM is due to similarities between the structure of the DA and the translation table. This process is missing from elementary GAs, which use a stochastic process to introduce mutations. REM is also superior to GAs from the standpoint of acceleration, which suggests only a few rounds of CCM are required in an actual experiment. Very recent simulations suggest that a novel acceleration algorithm can be developed which uses nucleotide "dopes" (see Chapter 6) compatible with commercial DNA synthesizers.

4.2.5 Acknowledgement.

This work was supported by NIH GM42645 and DOE DE-FG02-90ER20019. A.P.A. is supported by an NIH Biophysics Training Grant. We thank Dr. M.M. Yang at CyberGen Inc. for a critical reading of the manuscript and for general confirmation of REM behavior using independently written C code. We would also like to thank Dr. Dudley Herschbach for communicating this paper.

4.3 Epilogue

REM uses a feedback mechanism analogous to that operating in GAs, thus similar phenomena are likely to apply. When the REM

recombination operator is employed, schema (schemata) which appear with a certain probability in the acceptance set will appear with the same probability in the next test set. [Schema probabilities in REM may be calculated using the group probability equation 3.5. (Note that, like GAs, the shorter and less complex the schema the higher its probability is likely to be.)] Thus, fit schema will be enriched each generation. In fact, if a REM recombination operator is used in place of the crossover operator in a standard GA, GA performance is apparently not significantly impaired (data not shown).

The fitness function used by REM has no local optima *per se*; and a sequence is simply either acceptable or is rejected. Thus, REM's sequence space is a set of plateaus and flatlands. The goal of REM is not necessarily to find a global optimum, but to cover as many plateaus as evenly as possible. In practice, REM will generally cluster sequences on a single plateau (i.e. will find a population of highly related sequences). If REM is restarted with a different initial population then "convergence" to a different plateau is likely. Experimentally, then, it will be advantageous to perform a number of REM experiments in parallel or to split acceptance sets into independent entities for REM calculations. There may be a rational way of recombining populations from these different experiments in order to conserve complexity and avoid premature convergence to a clonal population. However, the theories for convergence which have been developed for GAs do not directly apply to REM; nonetheless many of the considerations which are necessary to achieve the optimal functioning of a GA have their counterparts in REM. For example, screening size (number of members in a GA population), selection criteria and epistasis (GA-hardness) are all important parameters which must be discussed in connection to REM's functioning. (See Chapters 5 and 6.)

The rest of this book is dedicated to the investigation of the speed. An important point must be discussed with regard to the extrapolative acceleration described in Chapter 4.2.3.6. The parameters, A and s , cannot be absolutely determined in advance of a REM experiment. If the numerical trend in base probabilities at a given site after the zeroth iteration is obvious, then a strong extrapolation may be applicable. On the other hand, when the trend is caused solely by "noise" introduced by sampling error or if the magnitude of the probability differences is small, then perhaps extrapolation should be foregone. The decision on which trends are significant can only be learned by the analysis of the base probability fluctuations resulting from purely experimental limitations. Since obtaining quantitative estimates of these statistics could involve more work than not using an acceleration at all, a degree of intuition will be necessary to choose the extrapolation parameters.

Finally, it should be noted that the dopes used in the simulations above were calculated to six significant figures. Obviously, in an experiment this resolution is unobtainable. Each phosphoramidite derivative of a nucleotide base, the substrates experimentally used to synthesize oligonucleotides, have different coupling efficiencies which are dependent upon the previous base used in the synthesis. This fact, in combination with other experimental errors (e.g. weighing errors, sequence-dependent transcription and mRNA degradation), will significantly reduce the resolution at which dopes may be constructed and expressed. Simulations have shown that REM performance is not unduly degraded as long as resolution is kept at or above 10% (data not shown). The effects of using dopes calculated at the resolution available with most DNA Synthesizers is discussed in Chapter 6.

The rest of this thesis is dedicated to the investigation of the many points raised in the above paper. The body of the next two chapters are concerned with the experimental implementation of REM. However, a number of theoretical points and extensions to REM will be addressed in the epilogues to these chapters.

1. Brand, J. G., M. H. Zehms and G. D. Rose. 1988. Protein Folding. *Ann. Rev. Biochem. Biophys.* 6: 167-171.

2. Doolittle, R. F., R. L. Limb and D. Finishing. 1991. A Method to Identify Protein Sequences That Fold into a Known Three-Dimensional Structure. *Science* 253: 164-170.

3. Doolittle, R. F. 1985. Protein Conformational Flexibility. *TIBS* 10: 299-303.

4. Fritzsche, M. E. and P. G. Wolynes. 1989. Toward Protein Folding Classification Hierarchies by Means of Restricted Memory Hierarchical Clustering. *J. Chem. Phys.* 91: 171-173.

5. Gaudin, W. F. S. 1988. The Role of Computer Simulation Techniques in Protein Engineering. *Prot. Eng.* 2: 9-12.

6. Givan, G. M. 1990. Protein Folding: Computational Approaches to an Evolutionary Problem. *Ann. Rev. Comput. Sci.* 4: 29-54.

7. Hill, A. J., P. Overington, M. J. Jorgensen and T. L. Blundell. 1989. The Computational Analysis of Sequence and Structures in Protein Folding and Design. *TIBS* 14: 209-210.

8. Jernigan, R. L. and M. Levitt. 1991. Accurate Prediction of the Rate of Site-Specific Folding of Site-Directed Mutagenesis on a Protein Core. *Methods* 25(2): 414-431.

9. Ruffolo, J. and D. Lewis. 1991. *J. Chem. Tech. Biotech.* 50: 108-110.

10. Gelger, J. R. and D. A. Graves. 1991. Biotechnology Bioengineering, and the Solution to Environmental Problems. *Appl. Biochem. Biotech.* 26: 211-212.

4.4 References

1. Prevelige, P. and G. D. Fasman. 1990. Chou-Fasman Prediction of the Secondary Structure of Proteins: The Chou-Fasman-Prevelige Algorithm. *In* Prediction of Protein Structure and the Principles of Protein Conformation. G. D. Fasman. (Ed.) (Plenum Press, New York) 391-416.
2. Fetrow, J. S., M. H. Zehfus and G. D. Rose. 1988. Protein Folding: New Twists. *Bio/Technology* **6**: 167-171.
3. Bowie, J. U., R. Luthy and D. Eisenberg. 1991. A Method to Identify Protein Sequences That Fold into a Known Three-Dimensional Structure. *Science* **253**: 164-170.
4. Fasman, G. D. 1989. Protein Conformational Prediction. *TIBS* **14**: 295-299.
5. Friedrichs, M. S. and P. G. Wolynes. 1989. Toward Protein Tertiary Structure Recognition by Means of Associative Memory Hamiltonians. *Science* **246**: 371-373.
6. Gunsteren, W. F. v. 1988. The Role of Computer Simulation Techniques in Protein Engineering. *Prot. Eng.* **2**: 5-13.
7. Reeke, G. N. 1988. Protein Folding: Computational Approaches to an Exponential-Time Problem. *Ann. Rev. Comput. Sci.* **3**: 59-84.
8. Sali, A., J. P. Overington, M. S. Johnson and T. L. Blundell. 1990. From Comparisons of Protein Sequences and Structures to Protein Modelling and Design. *TIBS* **15**: 235-240.
9. Lee, C. and M. Levitt. 1991. Accurate Prediction of the Stability and Activity Effects of Site-Directed Mutagenesis on a Protein Core. *Nature* **352**: 448-451.
10. Ratledge, C. and D. Lewis. 1991. *J. Chem. Tech. Biotech.* **28**: 109-110.
11. Geiger, J. R. and D. A. Graves. 1991. Biotechnology, Bioengineering, and the Solution to Environmental Problems. *Appl. Biochem. Biotech.* **28**: 811-812.

12. Houghten, R. A., C. Pinilla, S. E. Blondelle, J. R. Appel, C. T. Dooley and J. H. Cuervo. 1991. Generation and Use of Synthetic Peptide Libraries for Basic Research and Drug Discovery. *Nature* **354**: 84-86.
13. Smith, J. M. 1970. Natural Selection and the Concept of a Protein Space. *Nature* **225**: 563-564.
14. Hermes, J. D., S. C. Blacklow and J. R. Knowles. 1990. Searching Sequence Space by Definable Random Mutagenesis- Improving the Catalytic Potency of an Enzyme. *Proc. Natl. Acad. Sci. USA* **87**: 696-700.
15. Oliphant, A. R., A. L. Nussbaum and K. Struhl. 1986. Cloning of Random-Sequence Oligodeoxynucleotides. *Gene* **44**: 177-183.
16. Reidhaar-Olson, J. F., J. U. Bowie, R. M. Breyer, J. C. Hu, K. L. Knight, W. A. Lim, M. C. Mossing, D. A. Parsell, K. R. Shoemaker and R. T. Sauer. 1991. Random Mutagenesis of Protein Sequences Using Oligonucleotide Cassettes. *Methods in Enzymology* **208**: 564-587.
17. Reidhaar-Olson, J. F. and R. T. Sauer. 1988. Combinatorial Cassette Mutagenesis as a Probe of the Informational Content of Protein Sequences. *Science* **241**: 53-57.
18. Yang, M. M., W. J. Coleman and D. C. Youvan. 1990. Genetic Coding Algorithms For Engineering Membrane Proteins. *In* Reaction Centers of Photosynthetic Bacteria. M.-E. Michel-Beyerle. (Ed.) (Springer-Verlag, Germany) 209-218.
19. Youvan, D. C. 1991. Photosynthetic Reaction Centers: Interfacing Molecular Genetics and Optical Spectroscopy. *TIBS* **16**: 145-149.
20. Sjostrom, M. and S. Wold. 1985. A Multivariate Study of the Relationship Between the Genetic Code and the Physical-Chemical Properties of Amino Acids. *J. Mol. Evol.* **22**: 272-277.
21. Arkin, A. P. and D. C. Youvan. 1992. Optimizing Nucleotide Mixtures to Encode Specific Subsets of Amino Acids for Semi-Random Mutagenesis. *Bio/Technology* **10**: 297-300.

22. Kyte, J. and R. F. Doolittle. 1982. A Simple Method for Display the Hydrophatic Character of a Protein. *J. Mol. Biol.* **157**: 105-132.
23. Zamyatnin, A. A. 1972. Protein Volume in Solution. *Prog. Biophys. Mol. Biol.* **157**: 105-132.
24. Booker, L. B. 1989. Classifier Systems and Genetic Algorithms. *Artificial Intelligence* **40**: 235-282.
25. Davis, L. 1991. Handbook of Genetic Algorithms. Van Nostrand Reinhold, New York.
26. Goldberg, D. E. 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Publishing Company, Inc., New York.
27. Khinchin, A. I. 1957. Mathematical Foundations of Information Theory. Dover Publications, Inc., New York.
28. Tuerk, C. and L. Gold. 1990. Systematic Evolution of Ligands by Exponential Enrichment: RNA Ligands to Bacteriophage T4 DNA Polymerase. *Science* **249**: 505-510.
29. Rothwell, N. V. 1983. Understanding Genetics. Oxford University Press, New York.
30. Bylina, E. J., S. J. Robles and D. C. Youvan. 1988. Directed Mutation Affecting the Putative Bacteriochlorophyll-Binding Sites in the Light-Harvesting I of *Rhodobacter capsulatus*. *Isr. J. Chem.* **28**: 73-78.
31. Zuber, H. 1990. Consideration on the Structural Principles of the Antenna Complexes of Phototrophic bacteria. *In* Molecular Biology of Membrane-Bound Complexes in Phototrophic Bacteria. G. Drews and E. A. Dawes. (Ed.) (Plenum Press, New York) 161-180.
32. Coleman, W. J. and D. C. Youvan. 1990. Spectroscopic Analysis of Genetically Modified Photosynthetic Reaction Centers. *Annu. Rev. Biophys. Biophys. Chem.* **19**: 333-367.
33. Hanson, L. K., M. A. Thompson, M. C. Zerner and J. Fajer. 1988. Theoretical Models of Electrochromic and Environmental Effects on Bacterio-Chlorophylls and -Pheophytins in Reaction Centers. *In* The

Photosynthetic Reaction Center: Structure and Dynamics. J. Breton and A. Vermeglio. (Ed.) (Plenum Press, New York) 355-367.

The Effects of JA Stringency and Library Complexity on Seed

5.1 Forward: Experimental Limitations on REM

Recursive Ensemble Mutagenesis, as described in Chapter 4, assumed an experimentally reasonable library complexity (eight sites) and screening size (10,000-20,000 mutants). These variables were chosen somewhat arbitrarily such that more than two acceptable sequences were observed in the zeroth iteration of REM, regardless of which of the four DAs were employed. In practice, the number of proteins which may be reasonably screened is limited by the technology used to produce the library and examine each member. In our laboratory, cloning efficiency is currently limiting population complexity to approximately ten million and of these, approximately 10,000 may be screened per day by the Digital Imaging Spectrometer.

Another parameter arbitrarily chosen in the REM simulations is the stringency of the DA (i.e. how many proteins in the sequence space fit the DA criterion). The DAs used in Chapter 4 span a broad range of stringencies. In an actual protein the DA stringency may vary radically with the choice of doping region.¹ The following paper, which is submitted to the Parallel Problem Solving in Nature II Conference², attempts to investigate the effects of DA stringency and library complexity on the functioning of REM. The alphabetic DA, described in chapter 4, is employed not only because of its tunable stringency (it can allow different numbers of out-of-order characters) but also because it evaluates the fitness of a sequence based on a highly site-site dependent criterion and thus is expected to mimic much of the behavior of a DA one might encounter experimentally.

5.2 The Effects of DA Stringency and Library Complexity on REM

5.2.1 Summary

The effects of both decision algorithm (DA) stringency and library complexity on Recursive Ensemble Mutagenesis (REM) are examined by computer simulation. REM generally demonstrates maximum gain when the number of unique acceptable proteins observed in the zeroth iteration is very small relative to the screening size. This occurs under conditions of high stringency and moderate complexity.

5.2.2 Complexity and Stringency

Recursive Ensemble Mutagenesis is a method for engineering a diverse population of mutants expressing genetically altered proteins with desired properties. For a full exposition of REM, see Chapters 4 and 6. Experimentally, REM utilizes iterative rounds of combinatorial cassette mutagenesis (CCM)³⁻¹⁰ to maximize the number of *unique mutants* (UM). In this paper, we investigate conditions under which REM maximizes UM as a function of parameters involving protein library complexity and DA stringency.

For various protein engineering tasks, the number of possible sequences which can be examined is limited by practical constraints on the experiment. We define the *total* number of *unique mutants* (TUM) as the set of all mutants (or genetically altered protein sequences) acceptable by the DA. The ratio of TUM to the size of the sequence space being searched is defined as the *DA stringency*. Furthermore, the number of mutagenesis sites (within a protein sequence) and the degree to which these sites are initially randomized defines the *complexity* of the CCM library.

Many parameters which are important to REM (e.g. the population size necessary to conserve genetic diversity) are also important to the optimal functioning of genetic algorithms (GAs).^{3, 11} REM is most similar to GAs that use redundant coding.^{12, 13} However, theories¹⁴ which govern the choice of population size for GAs are not directly applicable to REM, because REM is not a GA.

5.2.3 Mathematical Background

In the calculations described below, an "alphabetic" DA is used. A sequence is acceptable if the one letter codes (Table 1 in reference 2 (Table 6.1 in Chapter 6)) for the amino acids appear in dictionary order, e.g. *AACDE* is acceptable, whereas *CACFE* is not. The stringency of this DA may be adjusted by allowing out-of-order amino acid pairs, e.g. *CACFE* has two out-of-order pairs (OOOPs): *CA* and *FE*. The alphabetic DA is biologically unrealistic, nonetheless it is informative, since we can calculate stringency analytically. For a sequence of length *L* composed of letters from an alphabet of cardinality *C* (21 in this case, 20 amino acids plus one stop command), there are:

$$N_C^0(L) = \binom{L+C-1}{L} \tag{Eq. 5.1}$$

alphabetically ordered sequences, where the 0 superscript specifies zero OOOPs. To calculate the number of sequences with one or more out-of-orders, we use the number of ordered sequences within the variable characters *m* and *n* (beginning and ending the alphabetic sequence, respectively). This number is given by:

$$N_{m,n}^0(L) = \begin{cases} 1 & L=1, n=m \\ \binom{(L-2)+(n-m+1)-1}{L-2} = N_{(n-m+1)}^0(L-2) & L>1 \end{cases}$$

Eq. 5.2

Then the number of alphabetically ordered sequences beginning with character m of an alphabet of cardinality C is simply the sum over all ending characters of Eq. 5.2:

$$N_{m/C}^0(L) = \begin{cases} m & L=1 \\ \sum_{n=1}^C N_{(n-m+1)}^0(L-2) & L>1 \end{cases}$$

Eq. 5.3

Finally, we can find the number of single OOO sequences by calculating the number of all ordered sequences of length $\lambda < L$, which begin with m and end with n , and then multiplying by the number of ordered sequences beginning with a character less than n . The resulting product is then summed over all possible beginning and ending characters and all $\lambda < L$.

$$N_C^1(L) = \sum_{\lambda=1}^{L-1} \sum_{m=1}^C \left[\sum_{n=1}^C N_{m,n}^0(\lambda) \sum_{p=1}^{n-1} N_{p/C}^0(L-\lambda) \right]$$

Eq. 5.4

To calculate sequences with two OOOs, one simply replaces the final sum over all sequences of length $L-\lambda$ by a another term similar to the one contained by the brackets in Eq. 5.4. This term is then summed over lengths from one to $L-\lambda-1$. For each additional OOO, one simply adds another sum.

The stringency of a DA is the sum of all the numbers of sequences having up to O OOOPs (i.e the TUM) divided by the library complexity. The smaller the resulting number, the more strict the DA. Table 2.1 shows stringencies for zero to five allowed OOOPs for sequence lengths of two to eight. Since REM usually starts with completely random DNA, the complexity of the protein library is 21^L .

The last thing to note is that the distribution of proteins resulting from a completely randomized DNA library (25% **A,C,G**, and **T** at each position in the DNA fragment) is inhomogeneous. This is a direct consequence of degeneracy in the genetic code.^{15, 16} Since leucine is assigned six times as many codons as methionine, it will appear more often in the protein library. For highly complex libraries, a reasonable approximation is to assume that all sequences appear equiprobably at a frequency of one over the theoretical complexity. In this case, one can calculate the chance of seeing any given sequence:

$$P_S(f) = 1 - (1-f)^S \quad \text{Eq. 5.5}$$

where $P_S(f)$ is the chance that one will see a specific protein sequence, which appears at a frequency f , after screening S sequences. Alternatively, since we assume that all the sequences appear at approximately the same frequency, $P_S(f)$ may be interpreted as the fraction of the library which has been observed after screening S sequences.

L \ O	0	1	2	3	4	5
2	0.5238	1.00	--	--	--	--
3	0.1911	0.8563	1.00	--	--	--
4	0.0546	0.5476	0.9692	1.00	--	--
5	0.0130	0.2706	0.8166	0.9950	1.00	--
6	0.0027	0.1091	0.5653	0.9426	0.9994	1.00
7	0.0005	0.0374	0.3232	0.7963	0.9855	0.9999
8	0.0001	0.0113	0.2237	0.6216	0.9413	0.9998

Table 5.1 This table lists stringencies for DAs allowing zero to five OOPs in sequences ranging from two to eight amino acids in length. The heading over each column is the number of OOPs allowed by the DA, and the label for each row is the length of the sequence.

5.2.4 Simulations

REM performance plots are calculated as described in reference 16, which is briefly discussed in the context of the REMNUC program in Chapter 6. For simulations on the effects of library complexity in REM, the stringency was set to one OOP and the length of the sequence was varied between two and eight amino acids. The upper limit for sequence length was set to eight, since this is compatible with experiments currently being performed in our laboratory. Simulations on the effects of rule stringency fix the sequence length to eight and vary the number of allowed OOPs from two to five. Figure 5.1 and Figure 5.2 show plots resulting from REM simulations on libraries of different complexities and stringencies, respectively. These data are compiled in Table 5.2 and Table 5.3.

One may estimate the expected UMs observed for the zeroth iteration of REM (for a given screening size) by using Equations 5.1, 4, and 5 (see Table 5.2). The largest initial amplification (1.65) and the maximum overall amplification (39.28) occurs when L equals

eight. In this case, we are screening a very small fraction (2.26×10^{-7}) of the total library. Only a small UM is initially observed. As shown in Figure 5.1, this number is rapidly amplified to more than half the total screening size.

For REM to initiate successfully, more than two different acceptable sequences must be found on the zeroth iteration. These plots demonstrate that when the DA becomes too strict (e.g. zero OOBPs), REM is unable to achieve a significant UM. At the opposite extreme, nonstringent DAs (e.g. five OOBPs) allow too many sequences through the zeroth iteration. At moderate stringency, REM demonstrates significant amplification and produces an experimentally desirable UM. Maximum amplification is achieved for the strictest DA, but a higher UM is achieved by less stringent DAs. Table 5.3 shows analytical and computer calculations for REM for DA stringency simulations.

Experimentally, it is important to estimate how library size, DA stringency, and sampling size interact. Using phage display libraries^{17, 18, 19, 20, 21} and peptide bead²² methods, up to 10^{10} sequences can be screened. One may wish to estimate how many amino acid sites can be simultaneously mutagenized using REM. If the experimental stringency for finding a functional mutant were similar to the stringency of a zero OOB DA, then the following equation can be maximized for L while maintaining a finite (>2) UM:

$$10^{10} \times \frac{N_{21}^0(L)}{21^L} = \text{UM} \geq 2 \quad \text{Eq. 5.6}$$

which yields $L = 14$.

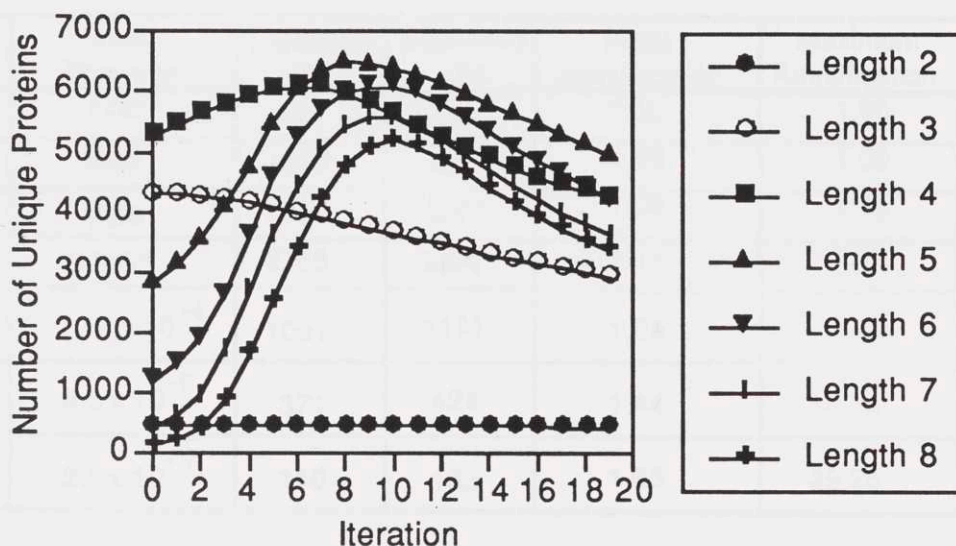


Figure 5.1. REM simulations using libraries of complexities ranging from 2^{12} to 2^{18} . The number of unique mutants (UMs) observed at each iteration of REM is plotted. Each plot is the average of ten simulations run with a different randomly generated initial population. The screening size is 10,000 and the DA allows one OOP.

Complexity	Fraction Screened	Expected UM	Observed UM	Initial Amplification	Maximum Amplification
441	1.00	441	440	1.00	1.00
9261	0.66	4839	4281	1.00	1.00
194481	0.05	5324	5257	1.04	1.16
4084101	0.0025	2763	2827	1.11	2.30
85766121	1.17×10^{-4}	1091	1191	1.24	5.10
1801088541	5.5×10^{-6}	371	424	1.44	14.14
3.78×10^{10}	2.6×10^{-7}	110	132	1.65	39.28

Table 5.2. This table contains numerical data relevant to plots shown in Figure 5.1 (screening size of 10,000). The complexity values (first column) are calculated by 21^L . The approximate fraction of the library screened, shown in column 2, is calculated by $P_{10,000}(1/21^L)$. The expected number of uniques accepted on the zeroth iteration is $P_{10,000}(1/21^L)$ multiplied by $[N^0(L) + N^1(L)]$. The column labeled "Observed UM" lists the number of acceptable proteins found during the zeroth iteration of the REM simulations. Finally, the initial and maximum amplifications are calculated by dividing the UMs observed in the first iteration and the UMs observed in the iteration of maximum throughput by UMs observed in the zeroth iteration.

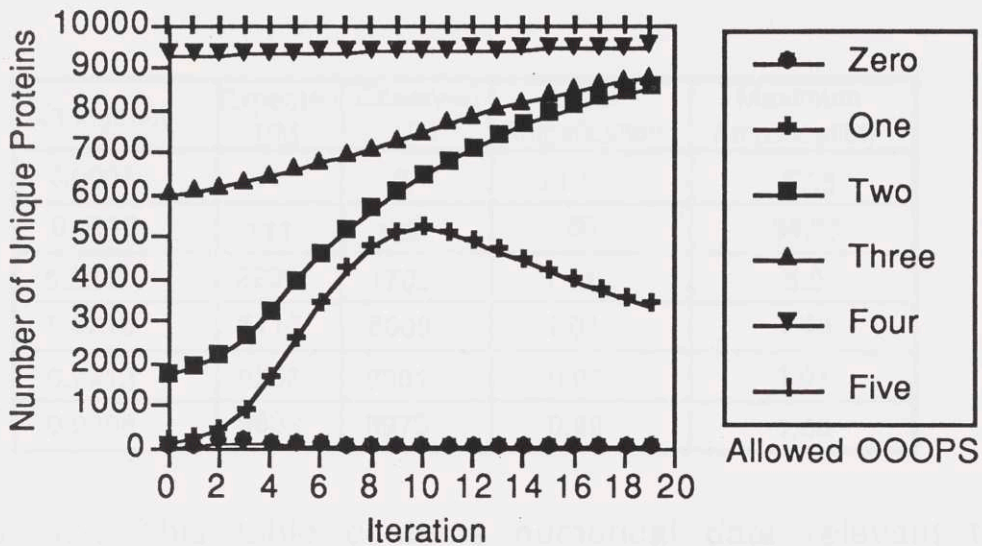


Figure 5.2. REM simulations using DAs which allow OOOPs to vary from zero to five. 10,000 sequences (of length eight) were screened per iteration. As in Figure 5.1, each plot is the average of ten independently seeded simulations.

Stringency	Expected UM	Observed UM	Initial Amplification	Maximum Amplification
0.0001	1	2	21.0	57.5
0.0113	111	132	1.65	39.29
0.2237	2200	1706	1.11	5.0
0.6216	6113	6009	1.01	1.46
0.9413	9256	9301	0.99	1.01
0.9998	9832	9975	0.99	1.00

Table 5.3. This table contains numerical data relevant to plots shown in Figure 5.2 (screening size of 10,000). The stringencies in the first column are identical to the stringencies appearing in the seventh row of Table 5.1. The approximate fraction of the library screened is 2.6×10^{-7} .

The results of the calculation using Eq. 5.6 suggest that one could simultaneously mutagenized 14 sites in a protein if the screening size were 10^{10} and the selection criteria were about as stringent as the alphabetic DA we have simulated. However, the stringency of the experimental selection criterion can only be determined by observing the UMs in the zeroth iteration of an actual REM experiment. If no positives are found, then the number of sites mutagenized (i.e. L) should be decreased.

5.2.5 Acknowledgments

This work was supported by NIH GM42645, and by an NIH Biophysical Training Grant to APA, supplemented by a Human Frontiers Science Program Award.

5.3 Epilogue

Perhaps this chapter raises more questions than it answers. First, how does one determine the stringency of the DA when it depends on which, and how many, sites are selected for mutagenesis? Further, for stringent DAs epistasis is expected to become important. How many parallel REM experiments must one carry out in order to probe this variable? Obviously, the most general answer to the first question is that the stringency must be observed by actually performing the experiment. But some estimate of the stringency might also be gleaned by examining phylogenetic and mutagenesis data. For example, a very rough approximation of the stringency for forming the β -subunit of Light Harvesting II of *Rb. capsulatus* might be calculated by assuming that every amino acid appearing in the phylogeny at a given site is allowed at that site

regardless of the amino acids appearing at every other site. (Note that this type of calculation is highly dependent on the number known members of the phylogeny.) If we calculate the approximate stringency for the sites discussed in Chapter 3, we find a stringency of 0.00018 (roughly $11,340/20^6$). So for the six sites mutagenized, approximately 23,000 proteins would have to be screened to observe four positive mutants. In reality, amino acids other than those which appear phylogenetically may be acceptable at each site and epistatic effects will govern which combinations of amino acids may appear simultaneously in the protein. The experimental stringency will, therefore, differ unpredictably from the above estimate.

If the observed stringency is too severe, then methods for "boot-strapping" into a population size acceptable to REM are required. The technique, described in Chapter 1.2, used by Geysen *et al.*²³ and Houghten *et al.*²⁴ may be applicable to this problem when translated for use with DNA. Essentially, this technique reduces the complexity of the library by breaking the problem into smaller pieces and optimizing each separately and then combining the results. Many variations on this scheme are possible but the final choice of protocol will likely be motivated by experimental results. Finally, it is possible in many cases to initially reduce the stringency of the screening in order to obtain an acceptable population size. The stringency may then be increased in subsequent iterations.

As a final note, an easy demonstration of epistasis can be given using the alphabetic rule. Consider the sequence "ACDETVWY" and suppose that three contiguous sites are to be randomized. If the first three sites are chosen, then the acceptance set for a zero OOP DA includes all ordered sequences whose first three positions form an ordered sequence composed of amino acids whose single letter

code is less than or equal to E (A,C,D, or E) and end with "ETVWY" (20 sequences). The constraint on the first portion of the sequence by the acids in the latter half and *vice versa* is an example of the epistasis mentioned in Chapter 4. If positions 3-5 had been mutagenized first, then the acceptance set is composed of ordered sequences composed of characters between D and V at that locus. The set is 48.45 times larger than that defined by the variation of the first three positions. Thus, if there is strong interaction between sites in a protein, the stringency of the DA can be very dependent on which sites are mutagenized. Dunn *et al.*'s experiments on the α -fragment of β -galactosidase clearly demonstrates this region specific stringency.¹

1. Dunn, S. C., Elowitz, H., Porter, and J. R. Knowles. 1998. A Facile Method for Random Mutagenesis: The Natural Use of Mutator Libraries Using Synthetic Oligonucleotide Primers. *Gene* 24: 149-151.

2. Elowitz, J. D., S. A. Elowitz, and J. R. Knowles. 1999. The Only Solution Space for Genetic Random Mutagenesis: Implying the Genetic Primacy of the Coding. *Proc Natl Acad Sci USA* 96: 616-620.

3. Dunn, S. C. and J. R. Knowles. 1994. The Role of Internal Packing Interactions in Determining the Structure and Stability of a Protein. *J Mol Biol* 243: 715-731.

4. Dunn, S. C. and R. T. Sever. 1989. Alternative Pathways of Mutagenesis in the Hydrophobic Core of λ repressor. *Proc Natl Acad Sci USA* 86: 111-115.

5. Morimoto, W. D. and H. C. Haughey. 1985. Targeted Random Mutagenesis: The Use of Ambiguously Synthesized Oligonucleotide Mutagenic Sequences Immediately 5' of an ATG Initiation Codon. *Nucl. Acids Res* 13: 3113-3121.

6. Heidhaar-Oleary, J. F. and R. T. Sever. 1988. Combinatorial Cassette Mutagenesis as a Probe of the Informational Content of Protein Sequences. *Science* 241: 53-57.

5.4 References

1. Dunn, I. S., R. Cowan and P. A. Jennings. 1988. Improved Peptide Function From Random Mutagenesis Over Short 'Windows'. *Prot. Engng.* **2**: 283-291.
2. Arkin, A. P. and D. C. Youvan. 1992. Recursive Ensemble Mutagenesis (II): Decision Algorithm Stringency and Library Complexity. *In* Parallel Problem Solving From Nature. B. Manderick. (Ed.) Submitted.
3. Davis, L. 1991. Handbook of Genetic Algorithms. Van Nostrand Reinhold, New York.
4. Hermes, J. D., S. M. Parekh, S. C. Blacklow, H. Koster and J. R. Knowles. 1989. A Reliable Method for Random Mutagenesis: The Generation of Mutant Libraries Using Spiked Oligo-Deoxyribonucleotide Primers. *Gene* **84**: 143-151.
5. Hermes, J. D., S. C. Blacklow and J. R. Knowles. 1990. Searching Sequence Space by Definable Random Mutagenesis- Improving the Catalytic Potency of an Enzyme. *Proc. Natl. Acad. Sci. USA* **87**: 696-700.
6. Lim, W. A. and R. T. Sauer. 1991. The Role of Internal Packing Interactions in Determining the Structure and Stability of a Protein. *J. Mol. Biol.* **219**: 359-376.
7. Lim, W. A. and R. T. Sauer. 1989. Alternative Packing Arrangements in the Hydrophobic Core of λ repressor. *Nature* **339**: 31-36.
8. Matteucci, M. D. and H. L. Heyneker. 1983. Targeted Random Mutagenesis: The Use of Ambiguously Synthesized Oligonucleotides to Mutagenize Sequences Immediately 5' of an ATG Initiation Codon. *Nucl. Acids Res.* **11**: 3113-3121.
9. Reidhaar-Olson, J. F. and R. T. Sauer. 1988. Combinatorial Cassette Mutagenesis as a Probe of the Informational Content of Protein Sequences. *Science* **241**: 53-57.

10. Reidhaar-Olson, J. F., J. U. Bowie, R. M. Breyer, J. C. Hu, K. L. Knight, W. A. Lim, M. C. Mossing, D. A. Parsell, K. R. Shoemaker and R. T. Sauer. 1991. Random Mutagenesis of Protein Sequences Using Oligonucleotide Cassettes. *Methods in Enzymology* **208**: 564-587.
11. Goldberg, D. E. 1989. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Publishing Company, Inc., New York.
12. Lopez, L. R. and H. J. Caulfield. 1990. A Principle of Minimum Complexity in Evolution. *In* Parallel Problem Solving From Nature. H.-P. Schwefel and R. Manner. (Ed.) 405-409.
13. Gerrits, M. and P. Hogeweg. 1990. Redundant Coding of an NP-Complete Problem Allows Efficient Genetic Algorithm Search. *In* Parallel Problem Solving From Nature. H.-P. Schwefel and R. Manner. (Ed.) 70-74.
14. Goldberg, D. E. 1989. Sizing Population for Serial and Parallel Genetic Algorithms. *In* Genetic Algorithms. J. D. Schaffer. (Ed.) 70-79.
15. Yang, M. M., W. J. Coleman and D. C. Youvan. 1990. Genetic Coding Algorithms For Engineering Membrane Proteins. *In* Reaction Centers of Photosynthetic Bacteria. M.-E. Michel-Beyerle. (Ed.) (Springer-Verlag, Germany) 209-218.
16. Arkin, A. P. and D. C. Youvan. 1992. Application of a Novel Technique to Protein Engineering: Simulation of Recursive Ensemble Mutagenesis. *Proc. Natl. Acad. Sci.* submitted.
17. Hoogenboom, H. R., A. D. Griffiths, K. S. Johnson, D. J. Chiswell, P. Hudson and G. Winter. 1991. Multi-Subunit Proteins on the Surface of Filamentous Phage: Methodologies for displaying antibody (Fab) heavy and Light Chains. *Nucl. Acid. Res.* **19**: 4133-4137.
18. Markland, W., B. L. Roberts, M. J. Saxena, S. K. Guterman and R. C. Ladner. 1991. Design, Construction and Function of a Multicopy Display Vector Using Fusions to the Major Coat Protein of Bacteriophage M13. *Gene* **109**: 13-19.
19. McCafferty, J., R. H. Jackson and D. J. Chiswell. 1991. Phage-Enzymes: Expression and Affinity Chromatography of Functional

Alkaline Phosphatase on the Surface of Bacteriophage. *Prot. Engng.* **4**: 955-961.

20. Roberts, B. L., W. Markland, A. C. Ley, R. B. Kent, D. W. White, S. K. Guterman and R. C. Ladner. 1992. Directed Evolution of a Protein: Selection of Potent Neutrophil Elastase Inhibitors Displayed on M13 Fusion Phage. *Proc. Natl. Acad. Sci. USA* **89**: 2429-2433.

21. Smith, G. P. 1985. Filamentous Fusion Phage: Novel Expression Vectors That Display Cloned Antigens on the Virion Surface. *Science* **228**: 1315-1317.

22. Lam, K. S., S. E. Salmon, E. M. Hersh, V. J. Hruby, W. M. Kazmierski and R. J. Knapp. 1991. A New Type of Synthetic Peptide Library for Identifying Ligand-Binding Activity. *Nature* **354**: 82-84.

23. Geysen, H. M., S. J. Rodda and T. J. Mason. 1986. *A Priori* Delineation of a Peptide Which Mimic a Discontinuous Antigenic Determinant. *Mol. Immunol.* **23**: 709-715.

24. Houghten, R. A., C. Pinilla, S. E. Blondelle, J. R. Appel, C. T. Dooley and J. H. Cuervo. 1991. Generation and Use of Synthetic Peptide Libraries for Basic Research and Drug Discovery. *Nature* **354**: 84-86.

Chapter 6

6.1 Forward

Towards a Practical Implementation of REM: REM and Optimized Nucleotide Mixtures

In Chapter 5, we have seen how the REM algorithm can be used to generate a set of nucleotide mixtures that are optimized for a given set of sequences. In this chapter, we will discuss how to use the REM algorithm to generate a set of nucleotide mixtures that are optimized for a given set of sequences. The first step is to generate a set of sequences that are optimized for a given set of sequences. This is done by using the REM algorithm to generate a set of sequences that are optimized for a given set of sequences. The second step is to generate a set of nucleotide mixtures that are optimized for a given set of sequences. This is done by using the REM algorithm to generate a set of nucleotide mixtures that are optimized for a given set of sequences. The third step is to generate a set of nucleotide mixtures that are optimized for a given set of sequences. This is done by using the REM algorithm to generate a set of nucleotide mixtures that are optimized for a given set of sequences. The fourth step is to generate a set of nucleotide mixtures that are optimized for a given set of sequences. This is done by using the REM algorithm to generate a set of nucleotide mixtures that are optimized for a given set of sequences. The fifth step is to generate a set of nucleotide mixtures that are optimized for a given set of sequences. This is done by using the REM algorithm to generate a set of nucleotide mixtures that are optimized for a given set of sequences. The sixth step is to generate a set of nucleotide mixtures that are optimized for a given set of sequences. This is done by using the REM algorithm to generate a set of nucleotide mixtures that are optimized for a given set of sequences. The seventh step is to generate a set of nucleotide mixtures that are optimized for a given set of sequences. This is done by using the REM algorithm to generate a set of nucleotide mixtures that are optimized for a given set of sequences. The eighth step is to generate a set of nucleotide mixtures that are optimized for a given set of sequences. This is done by using the REM algorithm to generate a set of nucleotide mixtures that are optimized for a given set of sequences. The ninth step is to generate a set of nucleotide mixtures that are optimized for a given set of sequences. This is done by using the REM algorithm to generate a set of nucleotide mixtures that are optimized for a given set of sequences. The tenth step is to generate a set of nucleotide mixtures that are optimized for a given set of sequences. This is done by using the REM algorithm to generate a set of nucleotide mixtures that are optimized for a given set of sequences.

Collaborator: Dr. Mary Yang

6.1 Forward

In Chapters 4 and 5, nucleotide dopes were constructed assuming practically unlimited resolution in setting the nucleotide probabilities. As mentioned in the epilogue to Chapter 5, there are physical limitations to the accuracy of the nucleotide mixtures used to synthesize a combinatorial cassette. Further, if REM were to be implemented on current DNA synthesizers a separate solution for each doped position would have to be made and the automated synthesis interrupted at the appropriate steps. This is a tedious process at best, thus, it may be best to employ the technology described in Chapter 3 to construct dopes consistent with the limited nucleotide solutions available on a standard machine. The following paper, which is submitted to the Parallel Problem Solving in Nature Conference II¹, is a discussion of REM implemented with a number of variations on these "intelligent" dopes.

6.2 REM Revisited

6.2.1 Summary

Genetic algorithms (GAs), which are based on classical genetics, are not easily converted from metaphors to useful algorithms for molecular geneticists. To work at the molecular level and engineer proteins *de novo*, we have developed a new combinatorial optimization technique, termed Recursive Ensemble Mutagenesis (REM). The experimental implementation of REM produces sets of proteins fitting an experimenter's selection criteria. As such, REM forms the basis of a new technology which could obviate the need for a solution to the protein folding problem, i.e. the prediction of structure and function of a protein from the

primary amino acid sequence. Here, we investigate (through computer simulation) optimization techniques and parameters that are fundamental to the experimental implementation of REM.

6.2.2 Introduction

Before 1980, geneticists used "classical" techniques to modify protein sequences. This technology relied on the chance that an uncontrolled mutation or recombination event might yield a mutant organism expressing a novel form of a protein that could be recognized through extensive screening and/or selection. With the advent of recombinant DNA technology and the capability to synthesize small fragments of DNA of a defined sequence (oligonucleotides), it became possible to alter a gene's sequence in a controlled and predetermined fashion.² Unfortunately, to design proteins *de novo* using such technology, one requires a solution to "the protein folding problem", wherein protein structure and function is calculated from the primary amino acid sequence.³⁻⁵ The difficulty associated with a solution to this problem is enormous, since the number of possible protein sequences grows exponentially with the length of the protein in amino acid residues. For a relatively small protein (e.g. length=100) there are 20^{100} possible sequences. [There are only about 10^{80} particles in the universe!] Using REM, we shall attempt to bypass the protein folding problem.

Oligonucleotide-mediated site-directed mutagenesis techniques that were developed in the early 1980's enabled geneticists to change only one amino acid residue in a protein at a time. More recently, combinatorial cassette mutagenesis (CCM) techniques have been developed⁶⁻⁸ which enable one to simultaneously change several residues. Using this latter procedure, a molecular geneticist can replace an entire segment of a gene

(encoding a specific domain within the protein) with a cassette of synthetic DNA. Since molecular cloning techniques can produce up to 10^{10} different recombinant mutants, the cassette's DNA sequence can be defined in a statistical fashion, such that many different combinations of mutations (i.e. amino acids) are present in the library. This depends in detail on how the DNA cassette is actually synthesized, i.e. various sites can be "doped" with DNA nucleotides (**A**, **C**, **G**, and **T** in different ratios).

Screening an entire CCM library becomes impossible as the number of mutagenized sites increases. If eight amino acid residues (not necessarily contiguous) are simultaneously mutagenized, over 25 billion possible protein sequences result. Since there are 20 naturally occurring amino acids (Table 1), this number increases by a factor of 20 every time an additional residue is mutagenized. Of these possible sequences, only a small fraction may actually be functional or possess properties which are desirable in the engineered protein. REM provides a new theoretical basis on which the information gained from sampling a small volume of this "sequence space" can be used in a recursive manner to isolate an ensemble of uniquely different proteins with desired characteristics.

The use of REM in the genetics laboratory is more than a computer simulation, since each step in the simulation has a direct experimental analog. The reality of conducting experiments establishes pragmatic constraints on the parameters and methods that are simulated. One can easily simulate the affects of changing simple parameters (e.g. sample size, or an error function used in optimization) that would otherwise require thousands of people-years of experimental labor.

Table 6.1 Genetic Code and Abbreviations

Amino Acid	3 letter	1 letter	Codons
Alanine	Ala	A	GCG, GCA, GCC, GCT
Arginine	Arg	R	AGA, AGG, CGG, CGA, CGC, CGT
Asparagine	Asn	N	AAT, AAC
Aspartic acid	Asp	D	GAT, GAC
Cysteine	Cys	C	TGT, TGC
Glutamine	Gln	Q	CAA, CAG
Glutamic acid	Glu	E	GAA, GAG
Glycine	Gly	G	GGG, GGA, GGC, GGT
Histidine	His	H	CAT, CAC
Isoleucine	Ile	I	ATT, ATC, ATA
Leucine	Leu	L	TTA, TTG, CTG, CTA, CTC, CTT
Lysine	Lys	K	AAA, AAG
Methionine	Met	M	ATG
Phenylalanine	Phe	F	TTT, TTC
Proline	Pro	P	CCG, CCA, CCC, CCT
Serine	Ser	S	AGT, AGC, TCG, TCA, TCC, TCT
Threonine	Thr	T	ACG, ACA, ACC, ACT
Tryptophan	Trp	W	TGG
Tyrosine	Tyr	Y	TAC, TAT
Valine	Val	V	GTG, GTA, GTC, GTT
Stop		X	TAG, TAA, TGA

6.2.3 Experimental Aspects of REM

DNA encodes the information to make protein by using 64 codons (or triplets of nucleotides) to encode 20 amino acids and one stop signal (Table 1). This 64 to 21 mapping ($3N \rightarrow N$, in length for DNA \rightarrow protein) indicates that the genetic code is degenerate. We have already discussed the experimental implications of a "structured" code, wherein amino acids with similar physicochemical properties are grouped in nonrandom ways^{9, 10}, using the degeneracy for added degrees of freedom in coding.¹¹

In the laboratory, a geneticist might pick several amino acid residues in a protein to subject to random or semi-random mutagenesis. Sites are picked based on expert rules: a) proximity to active sites shown by X-ray structures, b) phylogeny, c) previous

mutagenesis data, d) chemical modification data, e) chemical insight, etc. In picking the number of sites and the extent of randomization for REM, the experimenter must succeed in getting a few "positive" mutants out of a reasonable number screened in the zeroth iteration. Sequence data from the zeroth iteration are used to calculate intelligent nucleotide dopes¹² (see Chapter 3) for the next iteration of REM.

Selection or screening criteria are established to determine whether a particular CCM mutant is "positive". Positive mutants may include: 1) pseudo wild-types that are phenotypically (functionally) indistinguishable from the original wild-type, 2) mutants expressing higher levels of a particular activity, or 3) mutants expressing novel activities. Selection criteria are dependent on the protein of interest. For example, replica plating onto several sets of plates could be used to assay enhanced resistance against an inhibitor (e.g. antibiotic) or new activity against a different inhibitor. If a colorimetric indicator is available, then digital imaging spectroscopy (DIS) can be used as a rapid screening tool¹³⁻¹⁵. In addition, novel selection techniques have been recently developed that directly purify a "positive" protein linked to the DNA which encodes it (so-called phage display libraries¹⁶).

6.2.4 Overview of REM Methodology

REM is based on the recursive use of CCM which in turn relies on DNA synthesizers to generate DNA cassettes. Current day commercial DNA synthesizers are only capable of integer mixtures of subsets of the four nucleotides: pure **A**, **C**, **G**, or **T**; 1:1 mixture of any two; a 1:1:1 mixture of any three; or a 1:1:1:1 (equiprobable mixture, designated "**N**") of all four nucleotides. Since there are only 15 possible mixtures at any nucleotide site, there are $15^3 = 3375$

possible "synthesizer resolution" dopes for a triplet nucleotide (codon)¹²(Chapter 3). For example, if the following dope were entered into the DNA synthesizer:

G1	A1	T1	C1	G2	A2	T2	C2	G3	A3	T3	C3
0.5	0.0	0.0	0.5	0.0	0.0	1.0	0.0	0.0	0.0	0.5	0.5

we would expect **G ; C** each at a probability of 0.5 in the first codon position, **T** with a probability of 1.0 in the second codon position, and **T ; C** with a probability of 0.5 each in the third codon position. At the protein level, such a dope would generate leucine (*L*) and valine (*V*) with equal probability. [See Table 6.1 for the genetic code and single letter amino acid notation.]

A typical dope that has been used in CCM assigns equal probability to all four bases at each position in the codon. We define such a dope as "random": **NNN**. The other most commonly used dope is **NN(G,C)**. This is a more intelligent dope, since it is less complex at the DNA level (32 versus 64 codons) but still encodes all 20 amino acids. In order to take advantage of the enhancement properties of REM, the complexity of the possible peptide sequences arising from this CCM mutagenesis should be calculated and shown to be far in excess of the screening size.¹¹ (Chapters 4,5) Screening sizes are dependent on the difficulty of the molecular cloning techniques and the efficiency of screening or selection. At present, screening sizes can vary from 10^4 for difficult systems, where detailed information is obtained on each mutant, to 10^{10} where "on/off" phenotypes (e.g. phage display libraries binding to a column) are limited only by molecular cloning technology.)

The first step of experimental REM begins by expressing and screening a CCM library according to the rules described above.

Greater than two positive mutants are then picked and sequenced. Unless one is using "batch" or other rapid but non-standard sequencing methods, a maximum of 50 positive mutants is a reasonable number to pick, because DNA sequencing is a very tedious and time consuming job. Next, a list of unique protein sequences is determined by translating DNA sequences to protein. A "unique sequence" is defined at the protein level: if more than one protein has the same sequence, only the first occurrence of this sequence is retained and counted as unique. For each mutagenized position in the protein, a target set of acceptable amino acids is compiled from this list of uniques and the most appropriate dope is determined either by 1) the maximum group probability (p_G) or 2) the minimum sum of the squares of differences (SSD). These alternative algorithms are discussed in the following section on computer simulation. The next iteration of REM proceeds by using these intelligent dopes to generate the next population of mutants. All simulations in this manuscript are experimentally feasible since synthesizer resolution dopes are used.

6.2.5 Computer Simulations Using REM

Using very conservative experimental parameters, our computer simulations of REM employ a population size of 10,000, i.e. the maximum number of mutants to be screened in any iteration. For the zeroth iteration, we use a random dope of **NNN** for each codon. The DNA sequences are translated to protein using the biological genetic code and the protein is evaluated by a decision algorithm (DA). The DA is functionally analogous to gene expression and screening in the laboratory. DAs are a necessary construct in this simulation because of the inability to solve the protein folding and stability on the DNA sequences of "positive" mutants. REM/PEP decisions are directly analogous to actual laboratory procedures (undefined).

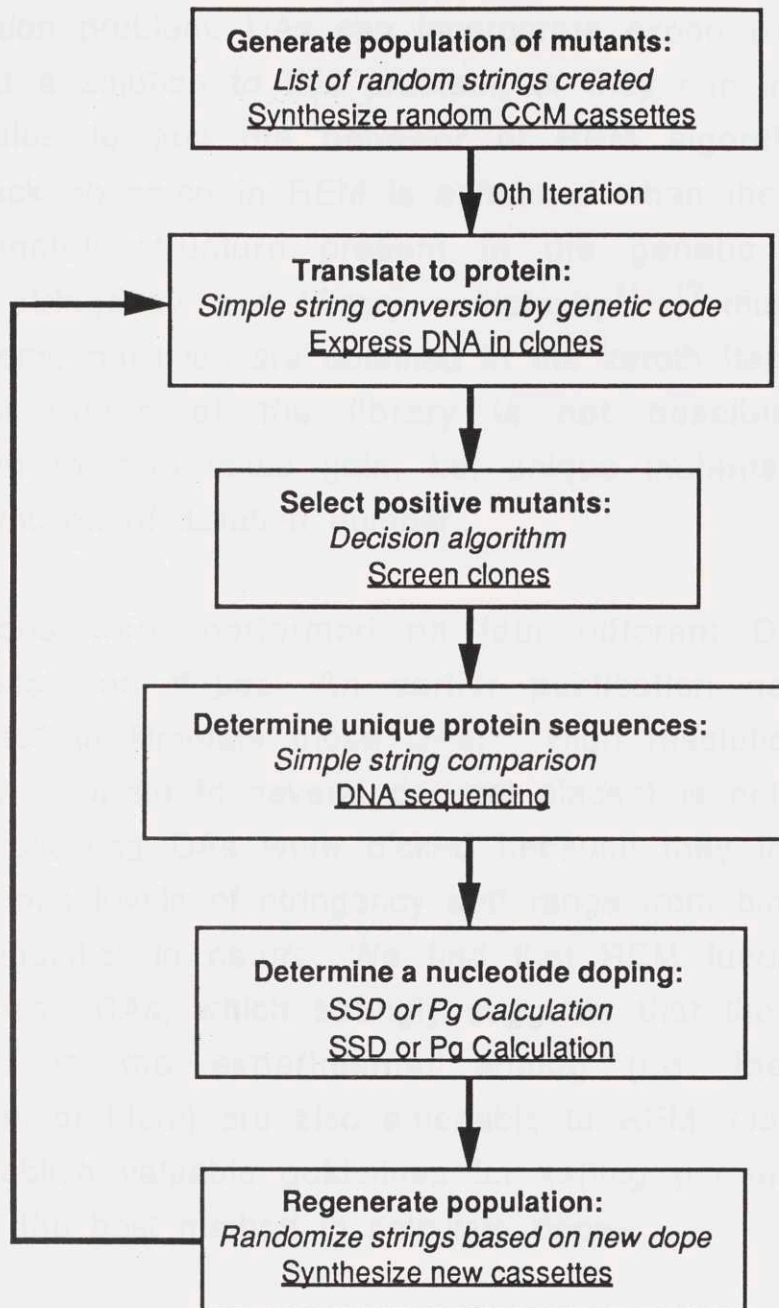


Figure 6.1. Flow chart of *computer simulated versus experimental REM*. The C-program REMPEP performs the operations shown in italics in this diagram. New dopes are based on an analysis of the deduced protein sequences of "unique" mutants. Another C-program (REMNUC) has been described^{11, 17} which calculates new dopes based solely on the DNA sequences of "positive" mutants. REMPEP simulations are directly analogous to actual laboratory procedures (underlined).

structure/function problem. DAs can incorporate expert rules which are guesses at a solution to this problem, or they can incorporate nonphysical rules to test the behavior of REM algorithms. The positive feedback observed in REM is enhanced when the DA uses criteria that match structure present in the genetic code. In addition, rule stringency and library complexity^{11, 17} must be in a range where some positives are obtained in the zeroth iteration, and simple over-screening of the library is not possible. These constraints lead to high initial gain, i.e. unique mutants increase rapidly as a function of iteration number.

Calculations were performed on four different DAs using synthesizer resolution dopes. An earlier publication used "high resolution dopes" to simulate these DAs.¹¹ High resolution doping (i.e. fractionally accurate to several decimal places) is not currently feasible. The following DAs were picked because they incorporate rules using various levels of stringency and range from biophysically realistic to "linguistic" in nature. We find that REM functions well under all of these DAs, which strongly suggests that the unknown "DAs" present in the experimental analog (i.e. the protein structure/function problem) are also amenable to REM. Our purpose here is to establish valuable guidelines for setting parameters and for determining the best method to calculate dopes.

1) *Alphabetic DA*: This decision algorithm takes as positive, any protein whose single letter amino acid codes are arranged in alphabetic order. This is a biophysically irrelevant rule which was chosen to demonstrate that the general behavior of REM is not peculiar to certain expert rules that might be applied to proteins.

2) *Alpha Helix DA*: This decision algorithm takes as positive, any octapeptide which has a high likelihood of forming an alpha helical structure as predicted by Chou-Fasman parameters.¹⁸

3) *Hydropathy DA*: This decision algorithm exploits the relationship between the genetic code and hydropathy (i.e. a molecule's tendency to hydrogen bond with water). For a given amino acid, this physical property may be predicted based on the A/T disparity in the second position of the codon. This DA operates on an octapeptide and constrains the hydropathic character of the peptide at selected positions. Hydropathy is known to be a very important factor in the structure and function of proteins.^{19, 20}

4) *Binding Site DA*: This is a complex DA based on expert rules that model possible sequence constraints on a protein binding a spectroscopically-shifted chlorophyll molecule. This DA includes specific constraints on hydropathy, molar volume, and protein grammar (deduced from phylogenetic and mutagenic studies^{21, 22}). Octapeptides are selected if they pass three requirements: a) one or two charged residues are present, b) the schema *axxxH* is present, where *a* is a small amphiphilic residue, *x* is any residue, and *H* is histidine, and c) the average hydropathy and molar volume of the octapeptide must be similar to wild-type sequence (*LAVLIHLL*) as previously described (Chapter 4).¹¹

In the REM simulation, a maximum number of 50 positives are picked. The unique protein sequences derived from these positives are then used to determine the nucleotide dopes for the next iteration of REM. Simulations were performed with 10 cycles (re-runs) of REM with 10 iterations each. Each cycle began with 10 independently seeded and randomly generated initial populations of 10,000 sequences. The C-program REMPEP outputs the *number* of

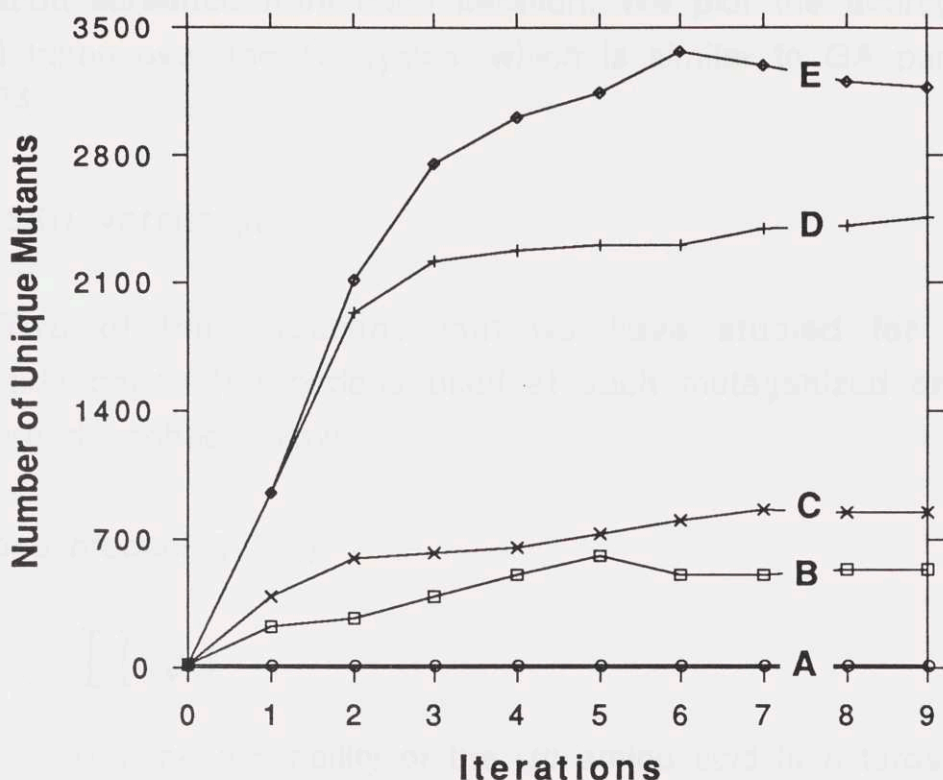


Figure 6.2. Number of unique mutants (UM) versus iterations of Recursive Ensemble Mutagenesis for 5 different simulations using the Binding Site decision algorithm. Qualitatively similar plots have been obtained for the other three DAs under a variety of parameters (extensive data not shown). In this figure, the following parameters were set constant: 1) Maximum screening size = 10,000, 2) Maximum number of positives selected to determine next dope = 50, 3) %Target = 100. Plot A was generated using **NNN** dopes for each iteration. The number of positives from 10,000 screened is less than 10 for all iterations. This is representative of what one might find if no positive feedback and/or non-intelligent dopes are used. Plot B uses p_G as defined by Eq. 6.1. Plot E uses SSD as defined by Eq. 6.2. Plot C uses equally weighted SSD terms (see text). Plot D uses SSD as specified by Eq. 6.2 and information from a running list of unique sequences from all earlier iterations.

positives (NP) and the number of *unique mutants* (UM) for the total population screened from each iteration. We plot the average of NP or UM taken over the 10 cycles, which is similar to GA performance plots.²³

6.2.6 SSD versus p_G

Two of the equations that we have studied for adjusting nucleotide dopes (for codons used at each mutagenized amino acid site) are described below:

1) Group probability (p_G):

$$p_G = \prod_i p_A(i) \quad \text{Eq. 6.1}$$

where $p_A(i)$ is the probability of the i th amino acid in a target set (i.e. a subset of the 20 amino acids) occurring based on a specific doping scheme.¹²

2) Sum of squares of the differences (SSD):

$$SSD = \sum_i (p_A(i) - p_T(i))^2 \quad \text{Eq. 6.2}$$

where $p_T(i)$ is the fractional representation of the i th amino acid (i.e. all 20).

The best dope is determined from the 3375 possible synthesizer resolution dopes by either maximizing p_G or minimizing SSD, depending on which method is being simulated. In cases where more than one dope satisfies the criterion, one out of the degenerate set of dopes is picked randomly. The calculation of each $p_A(i)$

accounts for the degeneracy of the genetic code and uses equations that we have previously described.¹²

An additional constraint that we have considered in both SSD and p_G analyses is to create a more robust target set by neglecting members that occur infrequently. This increases the likelihood that a dope more specific than **NNN** will be chosen, hence REM is "accelerated". This methodology is best shown by example. The members of the target set can be ranked in decreasing order according to their frequency of occurrence. Thus, seven unique mutants might yield three occurrences of glycine (**GGG**, **GGA**, **GGT** or **GGC**), three occurrences of alanine (**GCG**, **GCA**, **GCT** or **GCC**), and only one occurrence of proline (**CCG**, **CCA**, **CCT** or **CCC**) at one mutagenized position. Taking into account the ranking order, we define %Target to be the frequency of occurrence of one or more amino acids in the target set. This parameter can be used to adjust the diversity of the next generation. For example, a %Target of 100 would include all three amino acids. Whereas a %Target of 85 would only utilize codons encoding alanine and glycine. The resultant dope for this residue would contain **G** in the first codon position with 1.0 probability. Proline would never again be generated in subsequent iterations. By incrementing %Target and keeping other parameters constant, we find that the optimum value is 65 ± 10 regardless of the decision algorithm applied when using p_G (extensive data sets not shown).

If instead of ranking the occurrence of each amino acid in the target set, we gave each an equal weighting, we find that the number of positives and uniques for these calculations are lower than if we had simply used Eq. 6.2. This comparison is shown in Figure 6.2 in plots C and E, respectively. An explanation for this drop may be found in the structure of the genetic code and its codon

degeneracies. By giving each amino acid equal weighting, we do not take advantage of these inherent properties.

A similar argument can be made when comparing SSD and p_G calculations. From Eq. 6.1 and Eq. 6.2, we see that the frequency of occurrence is only taken into account by the %Target chosen when using p_G . On the other hand, SSD incorporates this information into the p_T term. This can explain the observation that SSD calculations are less sensitive to changes in %Target than p_G (extensive data not shown). Furthermore, simulations using SSD generated a higher UM by a factor of two or more for all four DAs discussed above.

Some REM simulations show UM increasing on the first few iterations, reaching a maximum, and then dropping to a clonal population at later iterations. The corresponding NP plot increases and then reaches a plateau at a value close to the number screened (data not shown). In a typical GA, one might vary the mutation rate or increase the frequency of the cross-over operator to generate more diversity. In REM, much of this functionality is determined by the translation code used, the codon degeneracies, and the independent scrambling of adjacent nucleotides. In an attempt to prevent the population from "going clonal", it is of interest to study the effects of compiling the unique mutant sequences from all previous iterations and using this information to determine the next dope. The result using SSD and a %Target of 100 is shown in Figure 2D. We find that UM and NP of iteration number two and above are less than UM and NP from similar calculations where previous iteration results were not incorporated. Although the population does not go clonal at higher iterations, many of the same mutants are generated and the number of uniques and positives level off.

In conclusion, we have found that the gain in the number of uniques screened is usually highest in the first iteration. Since our aim is not to converge to a single solution, but to acquire a diverse population of positives, our current results suggest that cycles of REM should be initiated in parallel and performed only a few iterations. In addition, the use of SSD equations (rather than p_G equations) appears to yield higher REM gains under a variety of DAs and parameters. This strongly suggests that SSD (Eq. 6.2) should be used in the experimental implementation of REM.

6.2.6 Acknowledgments

This work was supported by NIH GM42645, and by an NIH Biophysical Training Grant to APA, supplemented by a Human Frontiers Science Program Award.

6.3 Epilogue

Figure 6.2 shows a comparison of simulations performed with two different important REM recombination operators, REMNUC and SSD. This graph shows both the standard REM plots and a new type of plot derived from an important experimental parameter, Overall Uniques (OU). OU is the number of unique mutants obtained during a given iteration which have never been observed before. REMNUC (the operator described in Chapter 4) outperforms SSD both in the standard and OU plots though all plots converge, at high iteration, to a clonal population. The success of REMNUC may be attributed to its ability to conserve complexity at the nucleotide level (compare curve "○" to curve "□"). Both SSD and p_G rely on DNA Synthesizer resolution dopes, thus, base probabilities are often set to zero rather quickly. Further, SSD strongly minimizes the appearance of amino acids not observed in the acceptance set. This results in the

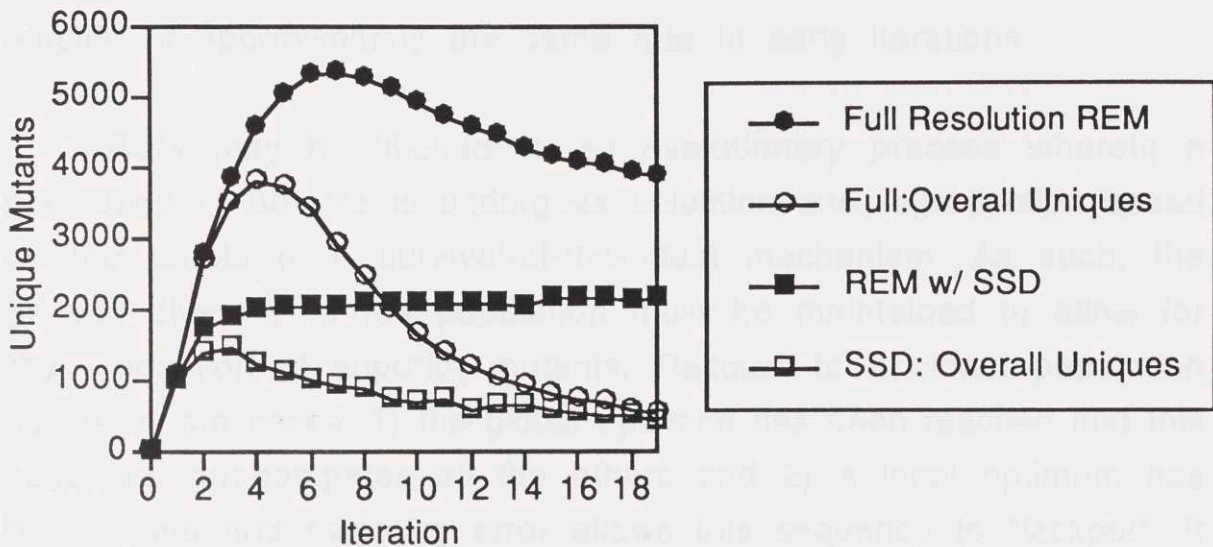


Figure 6.2 Comparison of Full REM and SSD. Each graph is a performance plot of REM simulations using a Bch Binding Site DA. For each recombination operator, SSD and REMNUC, two different strategies for counting unique mutants is employed: 1) The standard method which counts all the different sequences which appear during a given iteration or 2) another method which counts the number of different sequences obtained during a given iteration which have not appeared in any previous iteration (*Overall Uniques*). The plots are: ●) A performance average resulting from REM simulations at high resolution and using counting method 1 (see Chapters 4 and 6.2.5). ○) Same as ● but using counting method 2. ■) A performance average resulting from simulations using SSD operator and counting method 1. □) Same as ■ but using counting method 2.

production of fewer novel sequences through the cross-product mechanism. Practically, barring any unexpected advances in synthesis and cloning technology, only one or two iterations of REM will be experimentally performed. These iterations demonstrate the highest gain. All of the recombination operators described above amplify at approximately the same rate in early iterations.

REM may be likened to an evolutionary process wherein a population of organisms undergoes selection and reproduction based on the tenets of a survival-of-the-fittest mechanism. As such, the genetic diversity of the population must be maintained to allow for the production of superior mutants. Descent to a clonal population occurs in two cases: 1) the global optimum has been reached and this sequence out-competes all the others and 2) a local optimum has been found and sampling error allows this sequence to "jackpot". It is obviously desirable to avoid the occurrence of the second case, however, it is tempting to be satisfied with the results of the first case. But when the population is "evolving" in a changeable environment (e.g. in complex ecological systems or when the design criteria for protein engineering changes slightly due to new experimental results) then the former global optimum may well be demoted on the newly defined fitness surface. Clonal populations will not be able to adapt to the new constraints.

Therefore, when the design criteria are fuzzy (i.e. when they are subject to change or are ill-defined as is the case when determining whether a ground-state absorption spectrum is pseudo-wild-type or not), the goal of the recombination operator must be to maximize the average fitness of the population while assuring that diversity is maintained. A genetic algorithm conserves complexity, in part, through the random mutation operator. REM, which doesn't have a formal randomization operator, employs the cross-product

mechanism for the generation of novel sequences. Because REMNUC is slower to zero base-probabilities than SSD or p_G , cross-products are afforded a greater time to form and be observed.

The design of each of the REM operators was motivated by the experimental constraints of DNA synthesis, cloning, expression, and screening. However, there are many variations on the REM operators which have not been tried. The most obvious example is the inclusion of a simple random mutation operator which adds "noise" to a dope at some low rate. Random mutation at the DNA level is made difficult by the same factors as REMNUC; the resolution at which dopes can be constructed is limited by physical constraints. At the peptide level, however, the random inclusion of unobserved amino acids in the target sets used by SSD or p_G , may be a viable implementation which would conserve population complexity.

It is important to note, however, that REM succeeds in amplifying the number of unique fit sequences in all cases examined. This is a simple result of the recursive nature of the REM process. The actual amplification achieved by a REM experiment will be dependent on the nature of the sites chosen for mutagenesis and the recombination operator used to construct the nucleotide dopes. Each doped amino acid will have a different degree of interaction with other sites in the protein (due to either proximity or pleiotropic effects) and will contribute different amounts of epistasis to the feedback process. Therefore, REM trajectories may, perhaps, be a probe of amino acid interaction if the above effects are better quantified.

Further modification of the REM parameters will most likely be derived from actual experiments. This will certainly be the case when new methods are needed to "boot-strap" the initial population

into the REM regime as discussed in the epilogue to Chapter 5. In any case, the REM theory presents a number of testable hypotheses which must be verified by experiment.

Yasuda, M., and H. Yaman. 1992. Recursive Enumeration Algorithms for Combinatorial Optimization Techniques for Protein Engineering. *Abstracts: Methods Enzymol. 204*: 1-12.

Yasuda, M., M. Takahashi, H. Yaman, M. J. Zoller and M. Smith. 1992. *Abstracts: Proc. Natl. Acad. Sci.*

Yasuda, M., M. Takahashi, H. Yaman. 1993. Origin of Structure in Globular Proteins. *Proc. Natl. Acad. Sci.*

Yasuda, M., M. Takahashi, H. Yaman. 1991. Accurate Prediction of the Stability of Mutations Induced by Site-Directed Mutagenesis on a Protein Core. *Biopolymers* 24: 1-15.

Yasuda, M., H. Yaman, M. S. Johnson and T. L. Blundell. 1990. Relationship between Amino Acid Sequences and Structures in Protein Folding. *J. Mol. Biol.* 206: 235-260.

Yasuda, M., M. Takahashi, M. S. Johnson, R. M. Preyer, J. C. Ho, K. L. Knight, M. S. Johnson, G. A. Ford, H. G. Shoemaker and R. T. Sauer. 1991. Evolutionary Conservation of Protein Structure. *J. Mol. Biol.* 214: 565-587.

Yasuda, M., M. Takahashi, H. Yaman, R. M. Preyer. 1988. Combinatorial Analysis of the Information Content of Protein Sequences. *Biochem. Biophys. Res. Commun.* 154: 51-57.

Yasuda, M., P. A. L. Ruyter, H. Yaman, A. Kishi. 1986. Cloning of a Gene for the Gene Organization. *Gene* 44: 177-183.

Yasuda, M., and E. Wada. 1985. A Multifaceted Study of the Relationship between the Genetic Code and the Physical-Chemical Properties of Amino Acids. *J. Mol. Biol.* 22: 272-277.

Yasuda, M., W. J. Coleman and H. G. Yaman. 1990. Genetic Coding Algorithm for Engineering Membrane Proteins. In *Evolutionary Origins & Photosynthetic Bacteria*, M.-E. Michel-Beyerle (ed.) (Springer-Verlag, Germany) 209-218.

6.4 References

1. Yang, M. M., A. P. Arkin and D. C. Youvan. 1992. Recursive Ensemble Mutagenesis (I): A Combinatorial Optimization Technique for Protein Engineering. *In* Parallel Problem Solving From Nature. B. Manderick. (Ed.) Submitted.
2. Winter, G., A. R. Fersht, A. J. Wilkinson, M. J. Zoller and M. Smith. 1982. *Nature* **299**: 756-758.
3. Chan, S. H. and K. A. Dill. 1990. Origins of Structure in Globular Proteins. *PNAS* **87**: 6388-6392.
4. Lee, C. and M. Levitt. 1991. Accurate Prediction of the Stability and Activity Effects of Site-Directed Mutagenesis on a Protein Core. *Nature* **352**: 448-451.
5. Sali, A., J. P. Overington, M. S. Johnson and T. L. Blundell. 1990. From Comparisons of Protein Sequences and Structures to Protein Modelling and Design. *TIBS* **15**: 235-240.
6. Reidhaar-Olson, J. F., J. U. Bowie, R. M. Breyer, J. C. Hu, K. L. Knight, W. A. Lim, M. C. Mossing, D. A. Parsell, K. R. Shoemaker and R. T. Sauer. 1991. Random Mutagenesis of Protein Sequences Using Oligonucleotide Cassettes. *Methods in Enzymology* **208**: 564-587.
7. Reidhaar-Olson, J. F. and R. T. Sauer. 1988. Combinatorial Cassette Mutagenesis as a Probe of the Informational Content of Protein Sequences. *Science* **241**: 53-57.
8. Oliphant, A. R., A. L. Nussbaum and K. Struhl. 1986. Cloning of Random-Sequence Oligodeoxynucleotides. *Gene* **44**: 177-183.
9. Sjostrom, M. and S. Wold. 1985. A Multivariate Study of the Relationship Between the Genetic Code and the Physical-Chemical Properties of Amino Acids. *J. Mol. Evol.* **22**: 272-277.
10. Yang, M. M., W. J. Coleman and D. C. Youvan. 1990. Genetic Coding Algorithms For Engineering Membrane Proteins. *In* Reaction Centers of Photosynthetic Bacteria. M.-E. Michel-Beyerle. (Ed.) (Springer-Verlag, Germany) 209-218.

11. Arkin, A. P. and D. C. Youvan. 1992. Application of a Novel Technique to Protein Engineering: Simulation of Recursive Ensemble Mutagenesis. *Proc. Natl. Acad. Sci.* submitted.
12. Arkin, A. P. and D. C. Youvan. 1992. Optimizing Nucleotide Mixtures to Encode Specific Subsets of Amino Acids for Semi-Random Mutagenesis. *Bio/Technology* **10**: 297-300.
13. Arkin, A. P., E. Goldman, S. J. Robles, W. J. Coleman, C. A. Goddard, M. M. Yang and D. C. Youvan. 1990. Applications of Imaging Spectroscopy in Molecular Biology II: Colony Screening Based on Absorption Spectra. *Bio/Technology* **8**: 746-749.
14. Arkin, A. P. and D. C. Youvan. 1992. Digital Imaging Spectroscopy. *In* The Photosynthetic Reaction Center. J. Deisenhofer and J. R. Norris. (Ed.) (Academic Press, New York) In Press.
15. Yang, M. M. and D. C. Youvan. 1988. Applications of Imaging Spectroscopy in Molecular Biology: I. Screening Photosynthetic Bacteria. *Bio/Technology* **6**: 939-942.
16. Smith, G. P. 1985. Filamentous Fusion Phage: Novel Expression Vectors That Display Cloned Antigens on the Virion Surface. *Science* **228**: 1315-1317.
17. Arkin, A. P. and D. C. Youvan. 1992. Recursive Ensemble Mutagenesis (II): Decision Algorithm Stringency and Library Complexity. *In* Parallel Problem Solving From Nature. B. Manderick. (Ed.) Submitted.
18. Prevelige, P. and G. D. Fasman. 1990. Chou-Fasman Prediction of the Secondary Structure of Proteins: The Chou-Fasman-Prevelige Algorithm. *In* Prediction of Protein Structure and the Principles of Protein Conformation. G. D. Fasman. (Ed.) (Plenum Press, New York) 391-416.
19. Lim, W. A. and R. T. Sauer. 1989. Alternative Packing Arrangements in the Hydrophobic Core of λ repressor. *Nature* **339**: 31-36.
20. Youvan, D. C. 1991. Photosynthetic Reaction Centers: Interfacing Molecular Genetics and Optical Spectroscopy. *TIBS* **16**: 145-149.

21. Bylina, E. J., S. J. Robles and D. C. Youvan. 1988. Directed Mutation Affecting the Putative Bacteriochlorophyll-Binding Sites in the Light-Harvesting I of *Rhodobacter capsulatus*. *Isr. J. Chem.* **28**: 73-78.

22. Zuber, H. 1990. Consideration on the Structural Principles of the Antenna Complexes of Phototrophic bacteria. *In* Molecular Biology of Membrane-Bound Complexes in Phototrophic Bacteria. G. Drews and E. A. Dawes. (Ed.) (Plenum Press, New York) 161-180.

23. Davis, L. 1991. Handbook of Genetic Algorithms. Van Nostrand Reinhold, New York.

A.1 Post-Mortem

The following is a relatively brief annotated list of the most developed programs I wrote during my time at MIT. Many of the programs which are listed below are the result of aborted forays into unknown territory during lulls in the research described above. As such, these programs are undocumented and tend to be half-baked. They are listed here as a record for the laboratory and myself and are to be used as a library of useful subroutines and ideas for future research.

All the basic software for DIS, in both its video and CCD incarnations is listed herein. Perhaps the most important relevant annotation describes how to activate the CCD based DIS following compilation. Finally, a list of all the programs used to test REM and Optimized Nucleotide Mixtures is given. These include not only the actual REM and ONM programs themselves, but also programs which plot and analyze their output.

A.2 Annotated List of Programs

DIS/: Contains two implementations of DIS

CCD/ : Contains the Star I CCD implementation of DIS. All Dis programs should be compiled as in the shell "makeit". The resultant programs, dis.lock,dis.y,dis.info,main,ip,acq, and dsp should be placed in the dis home directory. Environmental Variables should be set as indicated in the DIS manual or as in the file ".login" in the /usr/people/adam account.

HomeDir/: Contains all files which must be present in the directory in which the DIS binary files reside.

IDFILE- File used to make internal tags in DIS.

blank- Contains values obtained for the blank area during the last run of DIS.

dummy/- An empty spectrum directory used to create new spectra.

params- Acquisition Parameters used during the lastrun of DIS.

slides- Filter Number to Wavelength Conversion.

times - Previous integration times.

Menus/: Source Code for the Menu, Buttons and Queries in DIS. Queries, Buttons and Menus use structures defined in includes/Menuing.h

ACQQueries.c- Questions asked by the acq (acquisition) program.

IPButtons.c- Buttons used in the ip.c program (image processing).

MainMenus.c- Menus use in the main.c program

DSPButtons.c- Buttons used by dsp.c (Display)

IPQueries.c- Queries used by ip.c

MainQueries.c- Queries used by main.c

includes/: Header files for the DIS programs. Many of these are duplicated in the library directories. (see below).

GeneralBuffers.h- Memory allocated for important data such as spectra, features, references, etc.

Menuing.h- Structures which define user interface gadgets.

hotmap.h- Color Map used by DIS for displaying spectra.

GlobalControl.h- Definitions for user interface variables.

libexternals.h- External definitions used by the DIS library.

FileManagement.h- Header file for programs utilizing the DIS file management routines (image and spectral files, acquisition parameters, etc.)

GpibControl.h- Header files for programs utilizing the National IEEE 488 (GPIB) controller for the Star I CCD camera.

SpectrumStructures.h- Structures to define a spectrum

GC.h- Header file used by the global control routines.

InterProcessControl.h- Defines variables necessary for DIS programs to communicate with each other. Used also for machine locking the binary code.

WindowManagement.h- Definitions for user interface.

library/: Contains all the source code to construct the DIS library

LFM/: File Management routines for reading and writing DIS data (spectra, images, parameters)

FileManagement.c- DIS library routines for data I/O.
FileManagement.h- Header file for programs using data I/O.

LGC/: IEEE 488 GPIB control routines: interface to Star I CCD

GC.h- Header files for GPIB control routines.
GpibControl.h- Header file for programs using GPIB
GpibControl.c- DIS library routines for GPIB control .

LGLC/: Interface to CSHELL environment

GlobalControl.c- Library for interface to unix environs.
GlobalControl.h- Header files for all DIS programs.

LIPC/: Interprocess communication routines. Used both for security reasons and so each program knows which other DIS programs are currently active.

InterProcessControl.c- Library routines for message I/O.
InterProcessControl.h- Header file for all DIS programs.

LM/: Holds basic routines for gadgets (text boxes, buttons, sliders, menus, queries, etc.) used by user interface.

Menuing.c- Library routines for menuing.
Menuing.h- Menuing definition for programs using gadgets.

LWM/: Window Management routines for determining window focus, SGI graphics environment, mouse actions, etc.

WindowManagement.c- Library routines for WM.
WindowManagement.h- DIS program definitions for WM.
hotmap.h- Color Map for TG series SGI computers.

program/: Spectrometer source code for image analysis, acquisition and display. Also includes machine locking facility.

dis.info.c- Program to get machine locking information.

dis.y.c- Program to output machine key given dis.info.

dis.lock.P- Program to activate newly compiled DIS with machine key.

After making DIS with "makeit" see below, type "dis.info | dis.y | dis.lock" in the DIS home directory. This will unlock DIS for use on a specific SGI machine.

ip.c- Image Analysis Program.

acq.c- Acquisition Program.

dsp.c - Classification and Display Program.

The above three programs can only be run by main.c.

main.c- Main control program. usage: main.

spectra/: Contains a spectrum (newtype) which runs with this version of DIS

utilities/: Contains utilities to be used only by DIS guru.

ConvertSToDRefs.c- Converts short format CCD DIS references files (files from pre-february, 1992) to a double format suitable for analytic absorption spectra.

c3130old.c- Converts SGI 3130 spectral data to FG-100 type 1 spectra.

makeit- Compile DIS library, DIS and all sub-programs.

ConvertShortToDouble.c- Converts short format CCD DIS spectra to the double format as stated above.

compile- Compile source to object code.

scvt.c- Converts type 1 FG-100 files to type 2 FG-100.

CON2.c- Convert type 1 (pre-june 1991) CCD DIS file to type 2.

cvt.c- Convert FG-100 type 1 image files to type 2.

CON3.c- Modified version of CON2 (use for files which CON2 seems to fail on.)

archive- Construct a library from object code.

doit- Compile and construct a library.

ConvertSToDBlanks.c- Converts short to double format for local refs.

bi.c- Automated Binarization program. Compile with:

cc -O -s -x -Llib bi.c -IDIS \$LIBS -o bi

usage: bi dis_image hysteretic_width

load- Optimize object files.

Video/: Contains IT FG-100 video camera implementation of DIS

FromHannoway/: Source code from GW Hannoway & Associates for driving FG-100 board.

dumpfg- Shell to display FG-100 registers.

initluts- Shell to init FG-100 color look-up tables.

ll.c- Source for look-up table control.

mem- Modification for memory map used by unix kernel.

snap- Shell to snap a picture on FG-100.

clear- Clear FG-100 frame buffers.

grab- Save FG-100 frame-buffer to disk.

io.c- Basic control routines for FG-100.

luts- Lut control shell.

save- Save FG-100 configuration.

HomeDir/: Contains files which must be present in DIS home directory.

DONTMOVEFILE- Used to generate shared memory tags in DIS.

dummyspec/: An empty spectrum used to create new spectra.

norm.gam- Gamma file for displaying images.

Source/: Source code for FG-100 DIS.

dis.c- The DIS source code....really horrendous!

hotmap.c- Color Map used by spectra display routines.

nrutil.c - Numerical Recipes memory management routines.

nrutil.h - Numerical Recipes memory management header.

popup.h - Popup menu definitions.

shmdefs.h- Shared Memory Definitions.

Compile DIS with:

cc -g dis.c \$LIBS -o dis

Spectra/: Contains spectra which are usable by this DIS implementation.

a28andrev4/: Reaction Center spectra.

splinter/: First Splintering poly-leucine mutants.

Utilities/: Utilities and tester programs for Video DIS.

c3130old.c- Convert SGI 3130 files to dis type 1 spectra.

cvt.c- Converts type 1 spectra to type 2 spectra.

fastdump.c- Dumps FG-100 data to disk.

idle- Turns off Slo-Syn Servomotor.

iread.c- Reads an image file.

program- Programs the Slo-Syn Servomotor Controller.

simil.c- Performs a similarity sort on type 2 spectra.

click.c- Sends signal to projector motion control.

dump.c- Another FG-100 dumping program.

fastio.c- Sends data to Slo-Syn Servomotor controller.

imgcvt- Shell to convert 3130 spectra to type 2 spectra.

iview.c- Displays images in a window.

scont.c- Displays a Map mode spectrum.

step- Causes Slo-Syn to move stepper motor one step.

cmak.c-

equalize.c- Performs simple base-line correction on type 2 spectra.

gamma.c-

imkgm.c-

makegam.c-

scvt.c- Converts a type 1 spectrum to a type 2.

adump.c-

f.c- Feature Extraction Algorithm.

home- Send Stepper-motor to its home position.

interp.c- Smooths spectra by polynomial interpolation.

on- Turn on stepper-motor.

seek- Moves stepper motor N steps from Home.

Doping/: Contains programs which investigate different methods for Combinatorial Mutagenesis.

OptNuc/: Contains Files necessary to reproduce the work in Chapter 3.

datafiles/: Data Files used by the "ent" programs (see below).

MACHINE- Machine Resolution Dopes.

NEWCODE- Genetic Code.

NEWHPMVS- Amino Acid Physical Properties.

P4.1P.25- High-Resolution Dopes.

src/: Source code for Optimized Nucleotide mixtures and plotting.

Dolt-

TestON.c- Newest optimized Nucleotide Mixture program.

nwentp.c- Analyze output from optnuc.c.

p1ent.c- Plot output from optnuc.c in H-M space.

pmt.c- Print master table from multiple nwent.c outputs.

subroutines/: Contains definitions for amino acids groups used by nwentp, p1ent, nwent, optnuc, pent, ppent.

subfilmway.c

subgfilmstvw.c

subnan.c

subntn.c

subfwy.c

subknrst.c

subncn.c

subsav.c

subagv.c

subga.c

subknst.c

subnnn.c

Sort- Sort output from nwent according to an objective function.

nwent.c- Another version of nwentp.c.

optnuc.c- Calculate Optimized Nucleotide Mixtures in H-M space.

pent.c- Plot optnuc.c output in H-M space.

ppent.c- Plot optnuc.c output in H-M space.

REM/: Contains Newest version of rem routines.

Clustering/: REM with heirarchical clustering. The purpose of this algorithm was to minimize the effects of epistasis caused by competing amino acid motifs. Acceptance sets were partitioned according to how they clustered on a heirarchically clustered tree. Similar Sequences should occupy the same branch of the tree. Thus, each partition should contain a single motif suitable for further REM amplification without epistatic interference. Preliminary results indicate that though epistasis is minimized, the overall diversity of the ensemble is drastically reduced.

Synonymy/: Contains data for amino acid synonymy which is used to calculate distances between sequences.

HPSyn- Hydropathy based Synonymy.

MPSyn- Molar Volume based Synonymy.

NSyn- Nucleotide Synonymy.

PSyn- Conservative synonymy.

data/: Data files used by REMhc (clustering REM).

DOP8- Eight Position Random Dope.

DaHoff.obs - The DaHoff amino acid substitution matrix.

HPMVS- Amino acid properties.

Seeds- Random Seeds for population generation.

WTCODE- The genetic code.

WTDNA- A wild-type BCH binding site.

src/: Source code for clustering REM.

Accept.c- Program to generate test set and acceptance set given DA.c.

OldBchDA.c- A BCH Binding Site DA.

Partition.c- Program to partion and acceptance set into n smaller acceptance sets. Used to test population splitting in REM.

Scramble.c- The REMNUC recombinations operator described in Chapter 4. Works on Output from Accept or Partition. Will do an extrapolative acceleration if asked.

Union.c- Combines results of partitioned REM simulations. Used to gauge overall diversity obtained by the splitting procedure.

partition- Shell for performing heirarchical partitions.

OptNuc.c- Calculate an optimized nucleotide mixture based on the acceptance set.

Report.c- Compiles a report on the efficacy of parallel REM simulations resulting from acceptance set partitioning.

SimpleShell- A shell to demonstrate how all the programs work together and to generate data on clustering effects.

rhc.c- A graphical heirarchical clustering program.

REM/: Contains the newest implementation of REM (May, 1992)
All REM programs compile with:

```
cc -O prog.c $LIBS -o prog
```

Usage is given by:

```
prog -u
```

datafiles/: Contains DataFiles used by the REM program.

ACODE- Alphabetic Translation Table.

CODESEEDS- Random Number Generator seeds used to make randomized translation tables.

Dampings- Values used to test the s parameter in the extrapolative acceleration routine.

SampleSizes- Values used to test the effects of screening size on REM.

WT- Wild-Type peptide for Bch Binding Site.

WTDNA- Wild-Type DNA for Bch Binding Site.

ALPHADNA- DNA for Bch Binding Site when the Alphabetic Translation Table is used.

DOP8- Eight Position **NNN** dope.

HPMVS- Amino Acid Physical Properties.

Slopes- Values tested for A parameter in the extrapolative acceleration routine.

WTCODE- The genetic code.

WildType- Wild-Type Bch Binding Peptide.

src/: Contains source code relevant to REM.

BchDA.c - Binding Site DA.
HAlphaDA.c - Zero OOP Alphabetic DA.
TestDA.c - Program to test decision algorithms.
arem.c - The REM program.
CFADA.c - Chou-Fasman Alpha-Helix DA.
OldBchDA.c- Standard Bch Binding DA.
TestON.c- Program to test if the optimized nucleotide mixture routines are working.
genseeds.c- Randomly generate seeds for a random number generator.
protein.dat- Header File for Chou-Fasman DA.
AlphaDA.c- 1 OOP Alphabetic DA.
RANDOM- Shell to test Random Translation Tables with REM.
TestUnq.c- Program to test whether the unique mutant counting routine is working.
jumb.c- Program to generate random translation tables.
shuffle.c- Program to generate shuffled translation tables.
AlphaSDA.c- Variable Stringency Alphabetic DA.
H2DA.c- Hydropathic DA.
SHUFFLE- Shell to test Shuffle Tables with REM.

EnergyMinimization/: Contains simple molecular energy minimization algorithms. These algorithms are very rough and were used to construct a hybrid genetic algorithm to be used in protein energy minimization.

GAmin/ contains a working prototype of the procedure with a very simplistic energy function. In effect, a population of protein structures are maintained. The genetic operators are:

1) random mutation- randomly change a structure based on a simulated annealing type criterion. The temperature is lowered every iteration of the GA.,

2) cross-over: Two structures are cut at a point on the protein backbone. Two new protein structure are constructed from swapping the resultant structure fragments and matching the dihedral angles at the point of recombination.

3) Conjugate gradient minimization may be performed on a given structure for a random number of steps.

Preliminary results suggest that this procedure is far more effective than conjugate gradients alone and may well be faster than simulated annealing. Nonetheless, since Charmm parameters are not implemented, rigorous testing was not performed. It is of interest however to analyze the relative contribution of the three genetic operators on the success of the algorithm. The best structures were obtained with a cross-over rate of 100%. Reducing the rate to below 50% severely reduced the efficacy of the algorithm. Since the kinetic pathway of protein folding is currently considered to occur by minimization of local regions in a peptide sequence followed by packing of the resultant secondary structures, it may be of interest to trace the evolution of the GA minimized structure to investigate if cross-over somehow packs locally minimized sub-structures.

ConjugateGradients/: Contains a simple conjugate-gradient minimization procedure.

cgmin.c- Conjugate gradient minimization program.

GAmin/: Contains files necessary for the hybrid GA minimizer.

datafiles/: Contains files used by gamin.c.

Params- Values for testing genetic recombination parameters.

include/: Header file for GA minimizer programs.

dbio.c - UCSD MMS database I/O routines.

mmsdef.h- UCSD MMS database definitions.

nrutil.c- Numerical Recipe memory management routines.

nrutil.h- Numerical Recipe memory management definitions.

molecules/: Test Molecules, in MMS format, used by gamin.c.

aminos.pic- Polypeptide using all amino acids.

pp.pic- Polypeptide of phenylalanine.

src/: Source code for GA minimizer and other algorithms.

PERFSHELL- Shell to investigate how recombination parameters affect minimization.

gamin.c - GA minimizer which plots ten best structures for a given iteration.

gaminT.c- GA minimizer which simply prints the energy of the ten best structures.

plotga.c- Plots results from PERFSHELL.

MolecularDynamics/: Contains the source code for a very simple molecular dynamics program. This program can use either Verlet, Leap-Frog, or Beeman algorithms for integrating the equations of motion and contains a very primitive self-consistent, electrostatic relaxation routine for modelling charging effects in proteins. This program is only a toy.

coul.c- Molecular Dynamics Program.

Compile: cc -O coul.c \$LIBS -o coul

For usage type "coul -u".

Photography: Contains programs which are useful for photographic data from SGI monitors.

center.c- Places a centering pattern on the screen so data may be accurately placed on the screen and the the camera will have a useful focus.

label.c- Makes a window containing a text label of any size and font rotated to any angle. This program is a bit buggy but it will work if one is patient.

mat.c- Makes a large borderless window displaying a specified color. Useful for making a neutral background placement of figures and such.

rulers.c- Places a vertical and horizontal ruler on the screen which may be used to align multiple figures.

ProteinModelling/: Contains random routines which may be used for the modelling of polymers or proteins. These are basically "toy" programs.

Lattice/: Contains programs which simulated a chain of beads with interaction between them on a lattice. The program was inspired by similar simulations carried out by Dr. K.A. Dill. The programs essentially use a simulated annealing algorithm to "fold" the chain into a low-energy conformation consistent with

the arbitrarily defined interactions. In their simplest form the programs were meant to investigate how well a protein could bury its hydrophobic amino acids and to determine the degeneracy of each of the packing arrangements. The programs are also designed to be able to deal with arbitrary lattice geometries. Lattice geometry may be restricted to simulate simple scaffolding which effects how the beads will fold. This research is motivated by the existence of chaperonins and the like which may determine how certain proteins fold.

lattice.c- A program to find low energy packings of a chain of interacting beads.

UnderDevel.c- Allows specified beads to be held stationary and marks the end of the bead chain.

MMS/: Contains a program built from the UCSD Molecular Modelling System which was used to determine the format of the MMS database.

db.c- Displays an MMS database file to a window.

dbio.c- MMS database I/O routines.

mmsdef.h- MMS database definitions.

SequenceAnalysis/: Contains programs for odd analyses of DNA or peptide sequences

Clustering/: Contains programs to perform heirarchical clustering on character sequences (DNA, Protein, or Alphabetic).

Synonymy- Conservative amino acid synonymy file.

rhc.c- Heirarchical clustering program.

Mutagen/: Contains Programs for finding and designing restriction sites in a DNA sequence.

CODE- Genetic Code.

CONSERVE- Definitions of conservative amino acid substitutions.

apl1- gene for alpha-subunit of the LH I poly-leucine mutant.

enzymes/: Contains lists of restriction enzymes and their restriction sites.

mp18uniq - Restriction Enzymes w/ unique sites in M13mp18.

neb - Restriction Enzymes Available from New England BioLabs.

pu27uniq- Restriction Enzymes w/ unique sites is plasmid pU27.

src- Programs for performing search and design.

conserv.c- Program to report: 1) all naturally occurring restriction sites, 2) restriction sites which may be created if silent mutations are acceptable and 3) restriction sites which may be created if conservative amino acid substitution is acceptable. Output may be sorted using the unix grep facilities.

res.c- Same as **conserv.c** but also reports non-conservative mutations.

SVD/: This directory contains programs which use the **SVDcov.c** program written by Dr. Mary Yang. The other programs are designed to analyze a set of related protein sequences, each of which has a value assigned to it based on its phenotype. The programs use a Singular Value Decomposition analysis to determine which amino acids at each site in the protein are most important in determining the phenotype of that protein. For example, if phylogenetic data from bacterial light-harvesting antennae is used, each sequence may be assigned a value of 1 or -1 for LH I-like and LH II-like spectra respectively. An SVD analysis should yield an estimate of which sites are most important in determining the absorption spectra. The problem is highly underdetermined in most cases since all possible proteins are not examined.

MakePhyloSVD.c - Takes a set of sequences in the format of **PHYLO2** (see below) and makes a file suitable for use with **SVDcov**. (See **PhyloSVD**.)

backcalculate.c- Uses the output from **SVDcov** and an SVD input matrix to back-calculate values which may be compared to the SVD solution vector (e.g. the phenotypic value vector mentioned above.)

data/: data used by **MakePhyloSVD** and **SVDcov**

PHYLO2- **MakePhyloSVD** input file example.

PhyloSVD- An output from **MakePhyloSVD**. May be used as input for **SVDcov**.

svdmvol.in- An SVDcov input file to test correlation of genetic code with molar volume.

SVDcov.c- Dr. Mary Yang's SVD analysis program.

SequencingGel/: Contains programs which simulate a simple sequencing gel based on end-labelled dideoxy-nucleotide DNA techniques. Designed to be a very simple calculation of relative band intensities based on sequence dependent effects.

dna- A sample DNA fragment.

seq.c- Sequencing gel simulator. (Toy program.)

Utility/: General utilities for /usr/people/adam

color.c- Changes a color in the color map.

nip.c- A modification of the SGI utility "ipaste". Allows images to be plotted as a photonegative.

printer.c- Basic program for interacting with the SGI font manager.

ramp.c- Create a color ramp in the color map.

sizeof.c- Prints the size of the standard C data types in bytes.

slidept.c- Plots graphs from REM data files. This has *many* options!