Research in Computational Astrobiology
Final Technical Report for Contract NCC2-5509

Galina Chaban,[a] Silvano Colombano,[b] Jeff Scargle,[c]
Michael H. New,[d] Andrew Pohorille,[d,e] Michael A. Wilson[d,e,f]

[a.] MS T27B-1, NASA Ames Research Center, Moffett Field, CA 94035
[b.] MS 230-3, NASA Ames Research Center, Moffett Field, CA 94035
[c.] MS 245-3, NASA Ames Research Center, Moffett Field, CA 94035
[d.] MS 239-4, NASA Ames Research Center, Moffett Field, CA 94035
[e.] Dept. Pharmaceutical Chemistry, University of California, San Francisco, CA 94143
[f.] Principal Investigator

Abstract

We report on several projects in the field of computational astrobiology, which is devoted to advancing our understanding of the origin, evolution and distribution of life in the Universe using theoretical and computational tools. Research projects included modifying existing computer simulation codes to use efficient, multiple time step algorithms, statistical methods for analysis of astrophysical data via optimal partitioning methods, electronic structure calculations on water-nuclei acid complexes, incorporation of structural information into genomic sequence analysis methods and calculations of shock-induced formation of polycylic aromatic hydrocarbon compounds.

Introduction

Computational Astrobiology uses computational and theoretical techniques to advance our understanding of the origin, evolution and distribution of life in the Universe. Computational astrobiology approaches these problems from several different points of view, ranging from the molecular and cellular level to the ecological and biosphere level. This requires exploiting information not only from the biological sciences, but also chemistry, geology, paleontology, and planetary and atmospheric sciences. Similarly, the goals of computational astrobiology cannot be accomplished using a single area of computer science but, instead, involve creative integration of several traditionally separate disciplines: biomodeling and biosimulations, bioinformatics, and complex systems science. Below we describe results from several projects that were carried out by a group of undergraduate and graduate students.

Electronic Structure Calculations on Nucleic Acid Bases. (Bridgit Crews and Galina Chaban) *Ab initio* studies of the spectroscopic properties of amino acids in the presence of water molecules have been performed to determine how formation of the complex with water affects the conformational structure and vibrational spectra of amino. The results of the study should be helpful in possible identification of these fundamental biological building blocks in space and/or interpretation of laboratory measured infrared spectra of amino acids. Complexes of amino acids with water will serve as a model of amino acids on small ice particles and mimic the conditions on icy grains in the interstellar medium.

Statistical Methods for Analysis of Astrophysical Data. (David Barnes and Jeff Scargle) Analysis of astrophysical data frequently involves extracting a signal from a noisy background. Algorithms are evaluated for optimal partitioning of a data set relative to a given objective function. Alternatives to the current approach of approximating the optimal partition are investigated to find a method that consistently returns a larger objective function value. The performance of other approximation algorithms, such as simulated annealing, are compared with that of the current method. A method to find the optimal partition of data in one dimension using dynamic programming has been found. An extension of this, or some similar process using dynamic programming, will be studied.

Microtubule-based Learning Models. (Jeffrey Pfaffman and Silvano Colombano) We are developing a microtubule-based learning model that combines a biologically motivated growth model with an abstract signaling dynamic. This system molds a virtual structure capable of producing a useful information transform. The model attempts to capture some of the essential aspects of how natural systems develop macromolecular structures, specifically cytoskeleton components, for adaptation to various environmental constraints. The information readin/out mechanism is based on an abstraction of microtubule associated proteins (MAPs) and introduces a competitive regulatory interface between the developed structure and some external device. This results in a dynamic learning mechanism that can alter the molded structure and, consequently, its information processing properties. We plan to test the feasibility of using this learning algorithm as a controller for a simple robotic system.

An inherited efficiencies model of non-genomic evolution. (Natasha Li, Michael H. New and Andrew Pohorille) As an alternative to the 'RNA World' hypothesis of the origin of life, we consider a non-genomic model of evolution in a 'peptide world' where catalytic functionality emerged prior to information storage. If we assume that catalytically efficient peptides have more ordered structures than less efficient peptides, they are less prone to be hydrolyzed, we can investigate the conditions under which the catalytic efficiency of such a system increases.

Structural bioinformatics. (Zhihui Xiao and Karl Schweighofer) We are interested in the practical application of structural analysis, database searching and design, molecular modeling, and structure prediction. To this end, we are interested in developing methods to incorporate structural information into sequence analysis and modeling methods, such as Hidden Markov Models, Support Vector Machines, and other supervised learning tools. One research area of interests to NASA is the phylogeny of essential molecular functions, such as ion channels, transporters, and signaling pathways.

Multiple Time-Step Algorithms in Molecular Dynamics. (Galen Reeves and Michael Wilson) To assist Ames' efforts in developing codes for molecular-level simulation of biological systems on massively parallel supercomputers, we are developing a multiple time-step code that will allow us to separate long and short-ranged forces. This code is implemented and tested with various long-ranged force methodologies (Ewald summation, Fast Multipole) and various separations of the long and short-ranged forces.

An Ab Inito Study: The effects of water complexation on the vibrational frequencies of glycine conformer 2

Bridgit Crews

Organisms are made up of a complex organization of unique proteins, which together perform a multitude of specialized tasks. Upon closer inspection of these seemingly intelligent mini-life forms one finds they are merely strings of smaller subunits in various orders and conformations. Due to this confoundingly simple arrangement of structure and sequence, life as we know it exists. These subunits are amino acids and necessarily, to gain better insight into the microscopic workings of living organisms we must look at the chemical nature of these pieces and what alterations they adopt under different environmental conditions. In spectroscopic studies it is possible to look at amino acids and peptides singularly without complicating solutions or averaging ensembles. However, by viewing amino acids or peptides singularly, and not in solution, we may lose some information about their nature in a normal biological environment. For example, it is known that a barrier to intramolecular proton transfer exists for glycine upon addition of solvent molecules and thus glycine zwitterion exists as a minimum in solution but not in gas. [4] In the present study we are interested in how complexation with water may shift the vibrat

tryptophan contain any polar groups in their side chains, so it is reasonable to believe that upon water complexation they will behave in very much the same way as glycine conformer 2.

Complexes of non-ionized glycine conformer 2 with water were studied with the goal to determine vibrational frequency shifts of glycine within the complex. Stationary point structures were located by ab inito methods using the electronic structure package package GAMESS. The geometries of the structures were optimized with no symmetry constraints at the level of restricted Hartree-Fock with second order MØller-Plesset (MP2) perturbation

Research in Computational Astrobiology
Final Technical Report for Contract NCC2-5509

Galina Chaban,[a] Silvano Colombano,[b] Jeff Scargle,[c]
Michael H. New,[d] Andrew Pohorille,[d,e] Michael A. Wilson[d,e,f]

[a.] MS T27B-1, NASA Ames Research Center, Moffett Field, CA 94035
[b.] MS 230-3, NASA Ames Research Center, Moffett Field, CA 94035
[c.] MS 245-3, NASA Ames Research Center, Moffett Field, CA 94035
[d.] MS 239-4, NASA Ames Research Center, Moffett Field, CA 94035
[e.] Dept. Pharmaceutical Chemistry, University of California, San Francisco, CA 94143
[f.] Principal Investigator

Abstract

We report on several projects in the field of computational astrobiology, which is devoted to advancing our understanding of the origin, evolution and distribution of life in the Universe using theoretical and computational tools. Research projects included modifying existing computer simulation codes to use efficient, multiple time step algorithms, statistical methods for analysis of astrophysical data via optimal partitioning methods, electronic structure calculations on water-nuclei acid complexes, incorporation of structural information into genomic sequence analysis methods and calculations of shock-induced formation of polycylic aromatic hydrocarbon compounds.

Introduction

Computational Astrobiology uses computational and theoretical techniques to advance our understanding of the origin, evolution and distribution of life in the Universe. Computational astrobiology approaches these problems from several different points of view, ranging from the molecular and cellular level to the ecological and biosphere level. This requires exploiting information not only from the biological sciences, but also chemistry, geology, paleontology, and planetary and atmospheric sciences. Similarly, the goals of computational astrobiology cannot be accomplished using a single area of computer science but, instead, involve creative integration of several traditionally separate disciplines: biomodeling and biosimulations, bioinformatics, and complex systems science. Below we describe results from several projects that were carried out by a group of undergraduate and graduate students.

Electronic Structure Calculations on Nucleic Acid Bases. (Bridgit Crews and Galina Chaban) *Ab initio* studies of the spectroscopic properties of amino acids in the presence of water molecules have been performed to determine how formation of the complex with water affects the conformational structure and vibrational spectra of amino. The results of the study should be helpful in possible identification of these fundamental biological building blocks in space and/or interpretation of laboratory measured infrared spectra of amino acids. Complexes of amino acids with water will serve as a model of amino acids on small ice particles and mimic the conditions on icy grains in the interstellar medium.

Statistical Methods for Analysis of Astrophysical Data. (David Barnes and Jeff Scargle) Analysis of astrophysical data frequently involves extracting a signal from a noisy background. Algorithms are evaluated for optimal partitioning of a data set relative to a given objective function. Alternatives to the current approach of approximating the optimal partition are investigated to find a method that consistently returns a larger objective function value. The performance of other approximation algorithms, such as simulated annealing, are compared with that of the current method. A method to find the optimal partition of data in one dimension using dynamic programming has been found. An extension of this, or some similar process using dynamic programming, will be studied.

Microtubule-based Learning Models. (Jeffrey Pfaffman and Silvano Colombano) We are developing a microtubule-based learning model that combines a biologically motivated growth model with an abstract signaling dynamic. This system molds a virtual structure capable of producing a useful information transform. The model attempts to capture some of the essential aspects of how natural systems develop macromolecular structures, specifically cytoskeleton components, for adaptation to various environmental constraints. The information readin/out mechanism is based on an abstraction of microtubule associated proteins (MAPs) and introduces a competitive regulatory interface between the developed structure and some external device. This results in a dynamic learning mechanism that can alter the molded structure and, consequently, its information processing properties. We plan to test the feasibility of using this learning algorithm as a controller for a simple robotic system.

An inherited efficiencies model of non-genomic evolution. (Natasha Li, Michael H. New and Andrew Pohorille) As an alternative to the 'RNA World' hypothesis of the origin of life, we consider a non-genomic model of evolution in a 'peptide world' where catalytic functionality emerged prior to information storage. If we assume that catalytically efficient peptides have more ordered structures than less efficient peptides, they are less prone to be hydrolyzed, we can investigate the conditions under which the catalytic efficiency of such a system increases.

Structural bioinformatics. (Zhihui Xiao and Karl Schweighofer) We are interested in the practical application of structural analysis, database searching and design, molecular modeling, and structure prediction. To this end, we are interested in developing methods to incorporate structural information into sequence analysis and modeling methods, such as Hidden Markov Models, Support Vector Machines, and other supervised learning tools. One research area of interests to NASA is the phylogeny of essential molecular functions, such as ion channels, transporters, and signaling pathways.

Multiple Time-Step Algorithms in Molecular Dynamics. (Galen Reeves and Michael Wilson) To assist Ames' efforts in developing codes for molecular-level simulation of biological systems on massively parallel supercomputers, we are developing a multiple time-step code that will allow us to separate long and short-ranged forces. This code is implemented and tested with various long-ranged force methodologies (Ewald summation, Fast Multipole) and various separations of the long and short-ranged forces.

An Ab Inito Study: The effects of water complexation on the vibrational frequencies of glycine conformer 2

Bridgit Crews

Organisms are made up of a complex organization of unique proteins, which together perform a multitude of specialized tasks. Upon closer inspection of these seemingly intelligent mini-life forms one finds they are merely strings of smaller subunits in various orders and conformations. Due to this confoundingly simple arrangement of structure and sequence, life as we know it exists. These subunits are amino acids and necessarily, to gain better insight into the microscopic workings of living organisms we must look at the chemical nature of these pieces and what alterations they adopt under different environmental conditions. In spectroscopic studies it is possible to look at amino acids and peptides singularly without complicating solutions or averaging ensembles. However, by viewing amino acids or peptides singularly, and not in solution, we may lose some information about their nature in a normal biological environment. For example, it is known that a barrier to intramolecular proton transfer exists for glycine upon addition of solvent molecules and thus glycine zwitterion exists as a minimum in solution but not in gas. [4] In the present study we are interested in how complexation with water may shift the vibrational frequencies of amino acids. This data may complement spectroscopic studies of amino acids or peptides in water clusters and may also prove helpful in identifying amino acids on ice grains in the interstellar medium.

During the initial studies that mapped the potential energy surface of glycine there was some discrepancy on whether non-ionized conformer 2 was indeed a minimum. This seemed to be a product of basis set choice and eventually it was concluded that the non-planar form of conformer 2 was indeed a minimum and lower in energy than the planar form.[1, 2, 3] Non-planar conformer 2 was chosen in this study because the orientation of the carboxyl and amino groups which form an intramolecular hydrogen bond are identical to that in the lowest energy conformers of phenylalanine and tryptophan. (Figure 1.1) [6, 7] Nether phenylalanine nor

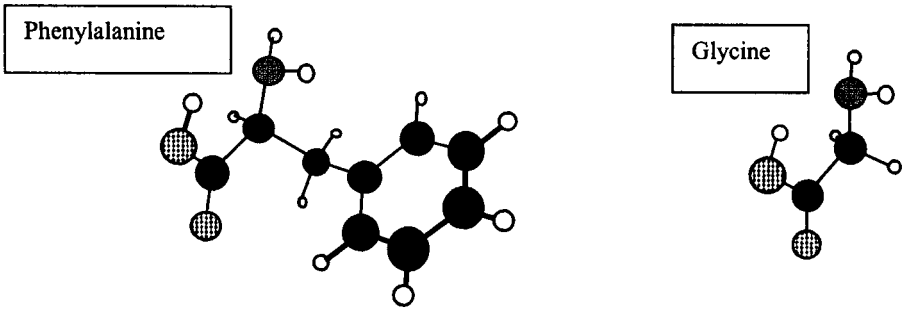


FIGURE 1.1

tryptophan contain any polar groups in their side chains, so it is reasonable to believe that upon water complexation they will behave in very much the same way as glycine conformer 2.

Complexes of non-ionized glycine conformer 2 with water were studied with the goal to determine vibrational frequency shifts of glycine within the complex. Stationary point structures were located by ab inito methods using the electronic structure package package GAMESS. The geometries of the structures were optimized with no symmetry constraints at the level of restricted Hartree-Fock with second order MØller-Plesset (MP2) perturbation theory. A Dunning-Hay double-ς + polarization basis set was employed, and polarized functions consisted of an additional d function on heavy atoms and a p function on hydrogen atoms. After stationary points were located the second-derivative (hessian) matrix was calculated to insure all frequencies were real and the points did indeed correspond to minima, not saddle points.

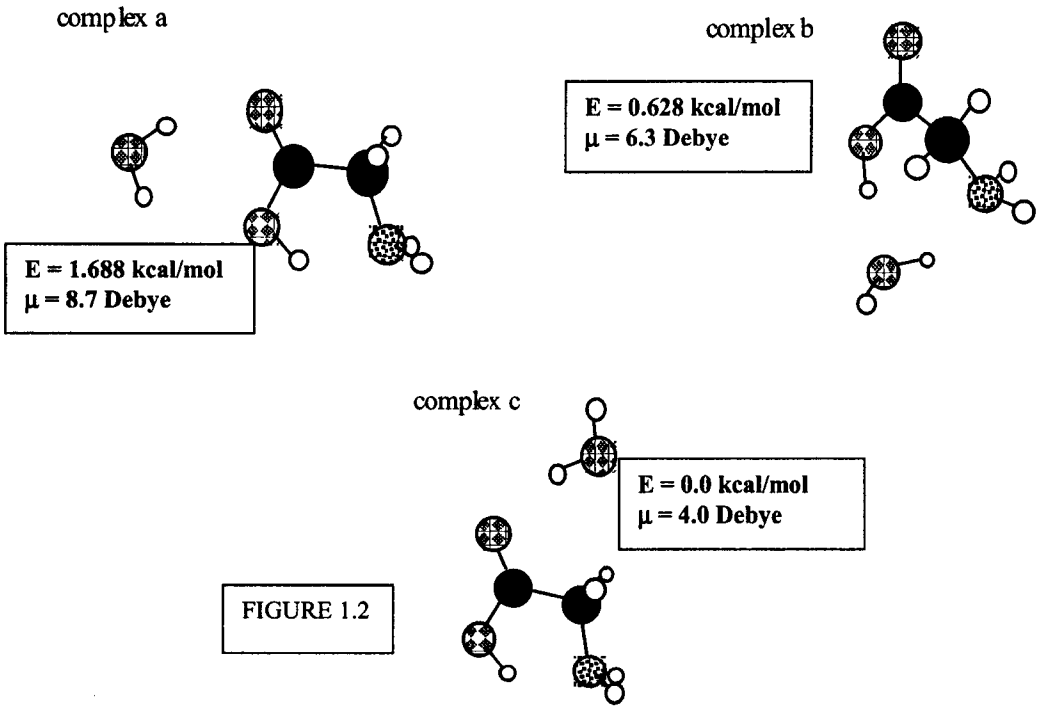


FIGURE 1.2

Three complexes of gly2 and 1 water molecule were chosen because they sampled the range of hydrophilic areas on gly2 while also being in position to form bridged structures, which are generally more stable than non-bridged structures. (Figure 1.2) Out of these three, complex c had the lowest intrinsic energy at –359.9515 Hartrees. The geometry of glycine in this complex is almost identical to free glycine, being 0.259 kcal/mol higher in energy. Complex b was only 0.6275 kcal/mol higher in energy than complex c, however its raise in energy is due entirely to its contorted form which is 6.796 kcal/mol higher than free glycine. It is the extremely high binding energy, -14.638 kcal/mol, which pulls the energy down to almost that of complex c. The binding energy of complex c is –8.731 kcal/mol, and that of complex a is –6.841 kcal/mol.[10] For this reason, it is possible that the barrier to de-complex is highest in complex b, and thus lifetime

and population of this form may be longer and greater than for either of the other two. However, confirmation of this speculation would require additional study of reaction paths leading to dissociation of the complex to glycine + H_2O.

The CO and OH stretching modes of glycine and the symmetric OH stretching mode of water showed frequency shifting upon complexation. Other modes (Table 1.1) also show shifting, but these modes are not nearly as intense and the shifting is not so systematic. Obviously, shifting is largely dependent on the position of H_2O, even so, each of the 3 modes listed above redshifts upon complexation. The OH stretch of glycine shows redshifts ranging from 67cm^{-1} to 129cm^{-1} with large relative IR intensities. The CO stretch shows redshifts ranging from 9 cm^{-1} to 33 cm^{-1}. The symmetric OH stretching mode of water by far shows the largest and most unique shifts in relation to position. Complex a shifts 32 cm^{-1}, complex c shifts 129 cm^{-1} ,and complex b shifts 353 cm^{-1}. In complex a, the water molecule is bound in a more symmetric manner, through both hydrogens, versus in the other two complexes where water binds asymmetrically through oxygen and hydrogen leaving a free hydrogen. This explains why the symmetric stretch would shift so much more in these complexes. For one-to-one complexes, it seems that the water symmetric stretching modes, which increase drastically in intensity upon complexation, should serve as the best identifiers of a particular complex conformation. The OH stretching frequencies of glycine also show unique shifts and large intensities and thus seem to be the most promising modes to look at in higher ratio complexes.

Next, the space of 2 H_2O + glycine (2:1) complexes was sampled. Three complexes with a water molecule in the position of complex b, and 1 with a water molecule in the position of complexes a and c were chosen. Complex b was chosen due to its high binding energy, and complex c was chosen because it is the lowest energy conformer. To these complexes, additional water

TABLE 1.1. Harmonic vibrational frequencies, cm -1 and IR intensities, Debye^2/amu-Angstrom^2

pure glycine pure water		**complex a**		glycine + 1 water **complex b**		**complex c**		description
harm	IR	harm	IR	harm	IR	harm	IR	
4056	1.1	4011	1.4	3982	2.2	4005	2.4	H2O asym stretch
3910	0.2	3878	2.6	3557	18	3781	6.6	H2O sym stretch
1666	2.1	1711	5.1	1688	1.4	1697	4	H2O bend
3714	0.4	3719	0.5	3666	0.3	3721	0.5	NH2 asym stretch
3612	0	3616	0.1	3573	4.1	3620	0.1	NH2 sym stretch
3559	7.1	3492	9.2	3430	12	3464	9.1	OH stretch
3225	0.2	3233	0.1	3212	0.4	3242	0.1	CH2 asym stretch
3152	0.3	3158	0.2	3136	0.5	3170	0.1	CH2 sym stretch
1860	6.9	1851	7	1827	5.8	1839	7.7	CO stretch
1446	9.6	1454	9.9	1415	5.7	1468	11	COH bend
1692	1	1693	1.1	1684	1.1	1690	0.4	NH2 bend
1507	0.2	1502	0.2	1541	0.1	1489	0.1	CH2 bend

molecules were added to previously sampled positions and also previously unsampled positions. (Figure 1.3) The naming of the 2:1 complexes corresponds to that of the 1:1 complexes such that complex AC contains a water molecule in the position of complex a and one in the position of complex c, and BB represents complex b with an additional water added in comparable b space. Complex B2 had the lowest energy at –436.2102 Hartrees. All relative energies are shown in Figure 1.3. All of the complexes containing water in position b remained contorted, only complex BB slightly unwound and glycine's intrinsic energy dropped by ~1 kcal/mol. In complex B2 the intrinsic energy of glycine rose by almost the same amount. Complex BB differs from B2 and all of the other 2:1 complexes because the second water molecule is bound to the oxygen of the first water molecule and the amine group of glycine, versus just glycine. This fact shows up interestingly in the frequency shifting, in that the OH stretch of glycine in BB is blueshifted by 149 cm^{-1} compared to that of complex b. This is an overall blueshift of 20 cm^{-1} from pure glycine.

This is most likely due to the inhibition of the NH stretching mode being locked in by the second H_2O (1). Because of this, the NH bond is not as responsive to the glycine OH stretching mode, which it interacts with through the OH of H_2O (2) thus, the OH stretch of glycine stiffens. Most other OH shifts remain red and increasing in size. (Table 1.2) The OH stretch of H_2O (2) (the initial water from complex b) shifts quite dramatically from pure water at 358 cm^{-1}, 441 cm^{-1}, and 543 cm^{-1}. This large shifting seems due to a slight correlation with the glycine hydroxyl group hydrogen that is absent in complex AC which only shifts 101cm^{-1}.

TABLE 1.2. Harmonic Vibrational Frequencies, cm-1 and relative IR intensities, Debye^2/amu-Angstrom^2 for 2 water complexes

complex AB		complex B2		complex BB		complex AC		description
4009	1.4	3998	2.5	3994	2.2	4013	1.2	H2O (1) asym str
3981	2.4	3979	2	3973	2.4	4007	3.2	H2O (2) asym str
3878	2.8	3812	3.1	3855	1.6	3873	3.1	H2O (1) sym str
3552	16	3469	30	3367	11	3809	5.3	H2O (2) sym stretch
1713	4.8	1707	1.2	1698	2.6	1712	5.5	H2O (1) bend
1680	1.9	1690	2.9	1664	2.4	1687	1.4	H2O (2) bend
3666	0.3	3636	0.8	3657	0.8	3721	0.6	NH2 asym str
3572	2.4	3527	0.9	3530	1.6	3621	0.1	NH2 sym str
3340	20	3339	9.7	3579	21	3400	11	OH (gly) stretch
3214	0.3	3209	0.4	3204	0.4	3246	0	CH2 asym str
3138	0.4	3137	0.6	3132	0.6	3174	0	CH2 sym str
1815	5.6	1809	5.9	1828	5.4	1829	7.5	CO stretch
1426	4.3	1436	4	1380	6.3	1473	11	COH bend
1687	1.2	1700	0.9	1693	2.6	1692	2	NH2 bend
1539	0.1	1543	0.2	1533	0.1	1486	0	CH2 bend

Compared to complex b, complexes AB and B2 show increased redshifts of the OH frequency of 90 cm^{-1} and 91 cm^{-1}, giving overall shifts from pure glycine of 214 cm^{-1} and 215 cm^{-1} respectively. AC shows a redshift of 154 cm^{-1} from pure glycine for this mode. The intensities of these modes also increase noticeably, more than twofold in some cases. The CO frequencies show smaller shifting from pure glycine in each case, at 45 cm^{-1} , 51 cm^{-1} , 32 cm^{-1}, and 31 cm^{-1} for AB, B2, BB, and AC respectively. It is interesting to note that in the case of complex BB the CO stretch does not show any further redshifting than its parent complex b except a 1 wavenumber blueshift.

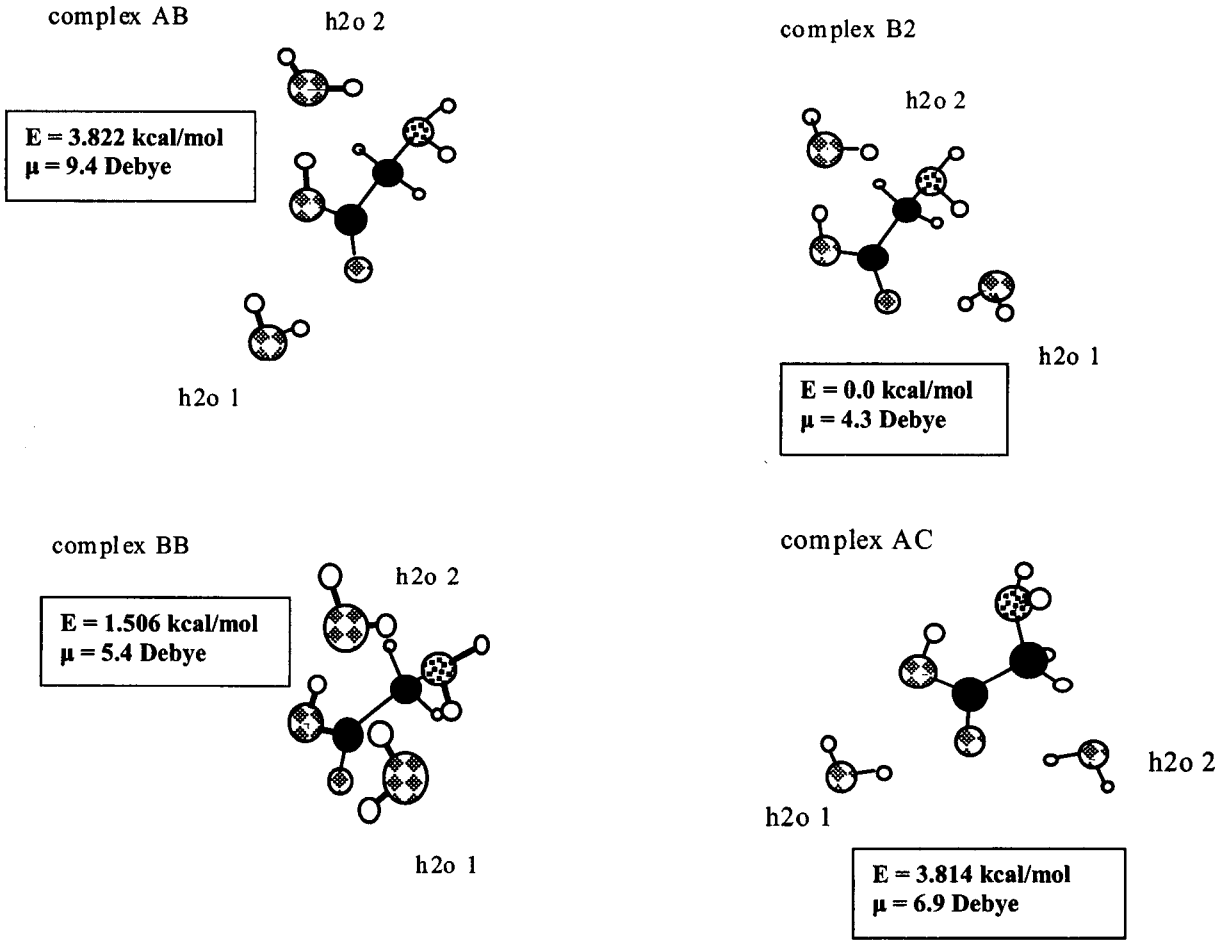


FIGURE 1.3

These results show that complexation has an observable, while not necessarily predictable, effect on stretching frequencies of glycine and water. How these effects will manifest in higher ratio complexes is uncertain. In the cases of water only being bound to glycine, redshifting was prevalent. However, when a second water molecule is bound to both glycine and an already existing water molecule some frequencies still shift more to the red while in others a halting and even reversal of this trend can be seen. Anharmonic effects are known to play a role in frequency shifting, especially in the case of hydrogen bonding. In previous cases of hydrogen bonding between glycine conformer one and water, anharmonic effects showed a redshift in H_2O stretching modes [9] and in previous studies on the OH stretching frequencies of glycine conformer 2, anharmonic effects also reveal a substantial redshift.[8] It will be necessary to asses this contribution for accurate analysis. Whether addition of more water molecules will inhibit the redshifting trend, or whether it will continue and the exception of a few blue shifting effects will be drowned out by the addition of more water molecules is a curious question that may be of interest in the aid of experiment. The next step in calculation will be to optimize 3:1, 4:1, and perhaps higher water: glycine complexes in order to sample the entire inner shell of solvent accessible area and glycine hydrophilic areas simultaneously. Also, by surrounding these higher complexes with additional layers of water molecules by described effective fragment potentials it will be possible to see how water complexation further affects the vibrational frequencies of glycine.

1 Csaszar A.G., J. Am. Chem. Soc. **1992**, 114, 9568-9575
2 Basch, H.; Stevens, W.J., Chem. Phys. Lett. **1990**, 169, 275-280
3 Jensen, J.; Gordon, M., J. Am. Chem. Soc. **1991**, 113, 7917-7924
4 Jensen, J.; Gordon, M., J. Am. Chem. Soc. **1995**, 117, 8159-8170
5 Lee, K. T.; Sung, J.; Lee, K. J.; Kim, S. D.; Park, Y.D.; J. Chem. Phys. **2002**, 116, 8251-8254
6 Snoek, L. C.; Robertson, E. G.; Kroemer, R. T.; Simons, J. P., Chem. Phys. Lett. **2000**, 321, 49-56
7 Snoek, L. C.; Kroemer, R. T.; Hockridge, M. R.; Simons, J. P., Phys. Chem. Chem. Phys. **2001**, 3, 1819-1826
8 Chaban, G. M.; Jung, J. O.; Gerber, R. B., J. Phys. Chem. A. **2000**, 104, 10035-10044
9 Chaban, G. M.; Gerber, R. B., J. Chem. Phys. **2001**, 115, 1340-1348
10 For a definition of binding Energy see ref. 4

Algorithms to find Regions of Uniform Rate in Poisson Data

David Barnes

1 Introduction

This work relates to research done by Jeff Scargle in finding signals in photon counting data with a noisy background. In the spring of 2002 he presented an optimization problem to a team of students working under Brad Jackson at San Jose State University. This problem was subsequently solved and a portion of this is a continuation of that work. The remaining part relates to the occurrence of the same problem in a higher dimension.

1.1 The Problem

The problem presented was to evaluate algorithms for finding optimal, or near optimal values for a given objective function. The function is used to find partitions of data in a bounded region of R^n. This analysis focuses on the cases when $n=1$ or $n=2$. These are the only two dimensions in which the function has been tried on real data. As the dimension of the space increases beyond two, visualization of partitions becomes more difficult. All of the algorithms and their implementations for the two dimensional problem will work equally well in higher dimensions, provided the integrity of the objective function is maintained in those higher spaces. A reasonable approximation of how well algorithms will perform in processing higher dimensional data is seen in the two-dimensional case.

Jackson and a CAMCOS (Center for Applied Mathematics and Computer Science) team at SJSU developed an algorithm that finds the global optimum for a given data set for the one-dimensional case. This is an $O(N^2)$ algorithm, which is an acceptable order for data sets of moderate size (less than 30,000 points). When processing larger sets, or when automating an analysis to evaluate a great number of sets, this becomes quite time consuming. Scargle has some groups of more than 10,000 data sets, most with more than 20,000 points. This motivates finding a faster version of that algorithm.

The algorithm operates at the lower bound for one capable of finding a provably optimal solution to the given problem, so no algorithm guaranteed to find the global optimum will have any better than $O(N^2)$ performance. Jackson developed an approximation algorithm that operates the same way as the original, but works on a smaller portion of the data, as determined by a variable input parameter. An analysis of that algorithm's performance for several values of the input will be made in section 2.

The above algorithm, or its approximation version, does not extend well to two or higher dimensions. Thus, other avenues must be explored when trying to improve on the original algorithm developed by Scargle. First, a more efficient implementation of the original algorithm was developed as a benchmark in evaluating other methods. Three alternative algorithms were

developed an implemented. The comparison of the algorithms for the two dimensional case will be made in Section 3. The parameter space for two of these has yet to be completely searched, so only preliminary results will be given as examples of their potential performance.

1.2 The Initial Division of the Space

To model a given data set, some initial partition of the space in which it lies is imposed as a starting point from which regions of nearly uniform intensity are built. This is accomplished by combining smaller partition sets, or blocks, to build larger blocks. Until recently, only one initial division, the Voronoi tessellation, of the bounded region B in which the data D occurs, had been tried in partitioning problems using this objective function.

Upon choosing B, or having it imposed by the device that recorded the data, an approximation of the intensity of a single point, p, can be made by finding the volume of the space consisting of all points in B closer to p than any other point in D. This is exactly the Voronoi tessellation of B, and amounts to generating intervals whose boundaries are the midpoints between consecutive data points in one dimension, and polygons that each contain exactly on point of D in two dimensions. For any dimension, the space that contains the point and all those points closer to it than any other point of D are referred to as Voronoi cells, or simply cells. These cells have a union equal to B, and pair-wise intersection equal to $\varnothing$ for distinct polygons. Since the problem is to create regions with approximately the same distribution of points of D throughout, using this initial partition seems quite reasonable.

An alternative exists though. The dual graph of the above tessellation is the Delaunay triangulation of the data. Points generating adjacent Voronoi cells are joined by an edge. The regions of interest now become the intervals between points in one dimension, and the triangles generated by this process in two dimensions.

When using the Voronoi tessellation problems can occur with cells that lie on the boundary of a region of high intensity, surrounded by a region of low intensity, or background noise. The volume for these cells will likely be greater than those cells that lie on the interior of this group. Hence, they may be interpreted by the objective function as not a part of the higher intensity group. If the Delaunay triangulation is used, this problem will not occur. Consider the one-dimensional case. If a high intensity region, I_1, occurs between two regions of relatively low intensity, I_2 and I_3, then the first and last points in I_1 will have Voronoi cells with large lengths relative to those completely contained in I_1. Using the Delaunay intervals instead, the two endpoints will still be associated with intervals of short length.

Associating points with cell when using the Delaunay triangulation can become problematic. In dimension one only two points may be associated with any interval, and each point may be adjacent to at most two intervals, so not much of a problem exists here. In dimension two though, each triangle is associated with three points, and each point may be adjacent to a number of triangles. In some simulations, it was not uncommon for nine or more triangles to meet at a single data point. On average though, one might expect to see four to five meeting at a point.

In higher dimensions it seems that it would be even more difficult to justify using Delaunay cells as approximations of local intensities. Therefore, the Voronoi tessellation was used as the initial partition of the bounded region in the analysis of all the algorithms. For all the algorithms, the initial partition is an independent process, and these two could be interchanged. For one-dimensional data, in some cases better results have been obtained by using Delaunay intervals. Thus the results given here might be improved by making this simple change. Regions created by combining Voronoi cells will be referred to as blocks, or Bayesian blocks as in several of Scargle's papers.

1.3 The Objective Function

The objective function is used to find the best partition, or model, for the data. It was derived using Bayesian statistics in an attempt to find a method to extract more information from a set of data than traditional methods. It does not necessitate binning the data, but does perform reasonably well on data that has been binned. The only requirement is that the points be of a Poisson process. The actual rate, or predictive rate of a given model, does not appear in the function. It is dependent only on the volume of the space in which a subset of points lie, and the number of points in the space.

Also referred to as a global posterior, the standard form of the function is:

$$p(m_n) = f(N,V) = \frac{\Gamma(N+1)\Gamma(V-N+1)}{\Gamma(V+2)} k\alpha^{-\eta}. \qquad (1)$$

It relates to the probability of a constant rate model m_n for a connected sub-region of B with volume V containing N points of D. The constant $k\alpha^{-\eta}$ accounts for a prior on the number of blocks in the complete model. In practice, it is used to eliminate small blocks of high intensity that are actually due to small fluctuations in the data unrelated to its true structure. This value is very application dependent and no concrete guidelines for choosing it are set at the present time.

For a data set of population n that lies in a region of volume v, the function value given by a model m of the data, where the space is divided into regions $r_1, r_2, \ldots r_k$, with corresponding volumes and populations $v_1, v_2, \ldots v_k$, $p_1, p_2, \ldots p_k$ respectively, such that $\sum_{i=1}^{k} v_i = v$ and $\sum_{i=1}^{k} p_i = n$, is given by

$$P(m) = \prod_{i=1}^{k} f(p_i, v_i). \qquad (2)$$

As there are often fairly large values for populations and volumes, it is far more practical to use the log of (1) in practice. So, any further mention of the objective function will be in reference to the following:

$$\ln(p(m_n)) = \ln(f(N,V)) = \ln(\Gamma(N+1)) + \ln(\Gamma(V-N+1)) - \ln(\Gamma(V+2)) + c \qquad (3)$$

and

$$\ln(P(m)) = \sum_{i=1}^{k} \ln(f(p_i, v_i)) . \qquad (4)$$

In (3) c represents the log of the prior on the number of blocks that appeared in (1). To illustrate worst-case performance of the algorithms, and since there are no formal justifications to support a choice of the value, c is left out, rather set to 0, in all the analyses to follow.

The problem now reduces simply to optimizing the objective function for a given set of data. The goodness of fit of the models can be decided by whoever is applying the technique. In one dimension, it actually reduces to finding the value of c best suits a particular application.

2 An Analysis of Algorithms for the One-Dimensional Problem

The problem in one dimension has a complete solution. Given a data set the algorithm developed by Jackson and the CAMCOS team will find the partition that yields the global maximum value for the objective function in $O(N^2)$ time. Jackson also developed an approximation algorithm that performs far better than any other approximation in terms of accuracy. The following compares the results given by several versions of the approximation and the original optimal algorithm.

2.1 Algorithms for the Problem in One Dimension

The original algorithm that provably returns the optimal solution to the partitioning problem is based on dynamic programming and the principle of optimality. This principle states simply that any sub-partition of an optimal partition is optimal. Sub-partition here refers to any subset of complete blocks in the optimal partition, not an arbitrary subset of the data. So any block, or subset of blocks, of the optimal partition is the best partition for the data contained within the block(s).

For the description of the algorithm, let $best(R)$ denote the known best partition of the first R cells. $best(0)$ is defined so that it may be used as a default term when considering the model that partitions all cells into a single block. Also, $end_i(j)$ denotes a partition of cells i to j, $i \le j$, into one block. It is assumed that the data is comprised of N distinct points (the case were some points are not distinct can be accommodated fairly easily with a slight change in step 1). In the following algorithm, and in any one-dimensional example here, the data points, and their cells, are labeled so point j is adjacent to point j+1. The algorithm is as follows:

Dynamic Algorithm, DA1

Step 1 Find the volume and population (v_j, p_j), associated with the j^{th} cell

Step 2 $P(best(0)) = 0$

Step 3 For $R = 1...\text{N}$ do

$$best(R) = model(i) \text{ such that}$$

$$P(model(i)) = \max_{1 \le j \le R} \{P(best(j-1)) + f(end_i(j)\}$$

Step 4 Return the partition $best(\text{N})$ and the value it yields for the objective function

For the approximation, the values of j are limited in step 3. Rather than continue to check every possibility, only those from some value x to R are checked, where x depends on an input defined at the beginning of the algorithm. While this algorithm is incomplete and counterexamples to it finding the optimal solution can be found, it can work fairly well, and in far less time than the other algorithm. Also, it is always focused on the portion of data that is most likely to change when a new point is added, those points closest to it. The variation in the input parameter allows one to decide how many of those points should be considered.

The term change point will refer to the boundary for a block as given by the algorithm. For DA1, no change points, or boundaries, are fixed until the algorithm terminates at step N. Previous to that last step some change points are likely to be fixed. If this happens for some change point, all those previous to it will be fixed as well. This is due to the principle of optimality and the method used by DA1. So while there does not seem to be a way of predicting such an occurrence, it seems likely that it will happen. This is the premise upon which this algorithm is built. The last few change points found at a certain stage might change at the next stage, but those that lie farther away from the new point being added to the process will probably stay the same. With this assumption, the algorithm above can be amended to only check for new change points between the new point and the k^{th} previous change point.

Since most data sets encountered in the application that generated the problem have few blocks, little more computation or storage is needed to consider the possibility that a previous change point might be the starting position of the last block at a given step. So, in addition to considering models that do not interact at all with change points that exist beyond a certain range, the objective function values for the new model involving old boundaries also computed. The approximation algorithm is as follows:

Dynamic Approximation Algorithm DAA

Step 1 Define parameter k = previous change point # the algorithm will check to.

Step 2 Find the volume and population (v_j, p_j), associated with the j^{th} cell.

Step 3 $P(best(0)) = 0$

Step 4 For $R = 1...\text{N}$ do

Let p_k be the point that is the k^{th} previous change point preceding point p_R

Also, let $c_1,...,c_{lcp}$ denote the change points up to, and including p_k, so $c_{lcp} = p_k$

$$best(R) = model(i) \text{ such that}$$

$$P(model(i)) = \max\left\{\begin{matrix} best(p_k) + \max\limits_{p_{k+1} \le j \le R} \{P(best(j-1)) + f(end_i(j))\} \\ \max\limits_{l \le j \le lcp} \{P(best(c_j)) + f(end_i(c_j))\} \end{matrix}\right\}$$

Step 5 Return the partition *best*(N) and the value it yields for the objective function

2.2 Comparison of DA1 with DAA for Three Values of *k*

Three versions of the approximation algorithm will be examined. Each one is faster than the original algorithm, so the problem at hand is to gauge which algorithm gives the best results when considering time and the expected difference from the true objective function value. Also to be taken into account are the exact locations of the change points. Often the precise boundaries of blocks are of interest. An algorithm that returns near-optimal function values, but rarely finds the true boundaries of the partition would not be desirable in that case.

To evaluate the performance of the algorithms data sets of several sizes were generated. This was done so the relationship between the time for the algorithms to terminate and the size of the data sets could be illustrated. For each of the ten sizes 1,000, 2,000, to 10,000, thirty data sets were constructed, 300 sets in all. All sets had an overall average rate (number of points/length of the interval in which they all lie) of approximately .05 points/unit. Every set was created using a random number generator included in the environment in which all the implementations were constructed and tested. The change points and structure found in this data is a product of points occurring unusually close together. With real photon counting data, or simulated data with a known structure, the results would likely be improved. Sharper, more consistent changes in the intensity of the data would yield more apparent boundaries between portions of the data. Even with such poorly conditioned data, it will be shown that the approximations are fairly good.

The optimal solution was found for each set, and the solution returned by each algorithm was compared with it. Averages for time, error (optimum function value – approximation function value), optimal change points found, optimal change points missed, and false change points found were computed for each group of thirty data sets. Most of the comparisons are based on these averages.

The greatest concern is whether or not the approximations are finding solutions that are at all close to the true global optimum. Figure 2.1 shows that as the parameter *k* is increased, the error decreases. No definitive trend is seen in any of the approximations, though it appears the average error grows slightly for each version of the algorithm as the data grows larger. The objective function values decrease, increase in absolute value, as the size of a data set increases. So the average relative errors for the data would likely be even better. As a note, the errors also became concentrated much more closely to the mean error as the value for *k* was increased.

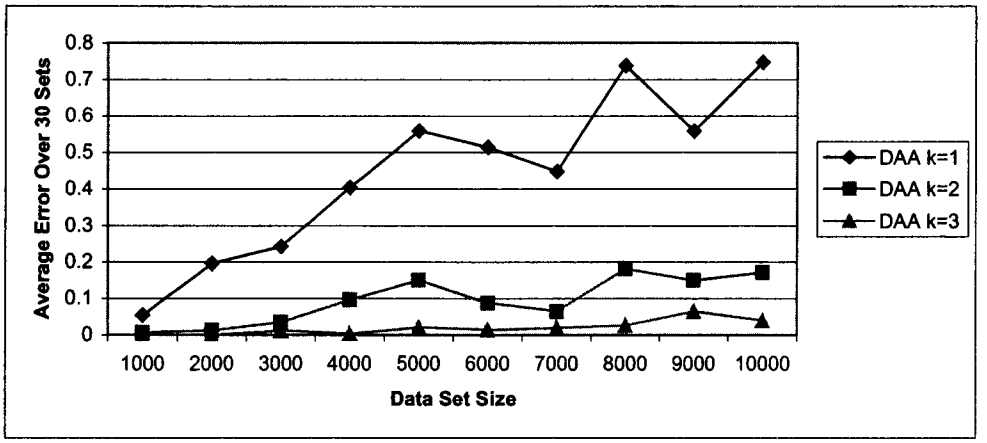


Figure 2.1 Average Errors for 30 Data Sets of Each Size

Given that all of the approximations seem to give reasonable results in terms of the value for the objective function, the next step is to consider how well the approximations find the true change points in the partition. First, notice in figure 2.2 that the number of change points in the optimal solution grows at approximately a linear rate as the size of the data increases. Similar growth in the average number of these true change points each approximation algorithm finds can be seen as well.

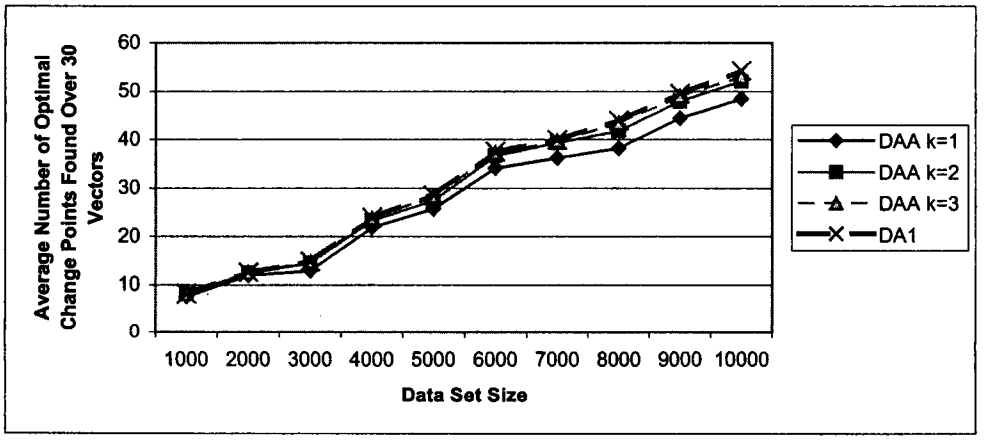


Figure 2.2 Average Optimal Change Points Found by Each Algorithm

So it seems that each approximation finds most of the change points most of the time. A better picture of how each approximation is performing in this arena can be seen in figures 2.3 and 2.4, which show average missed and non-optimal change points found, respectively.

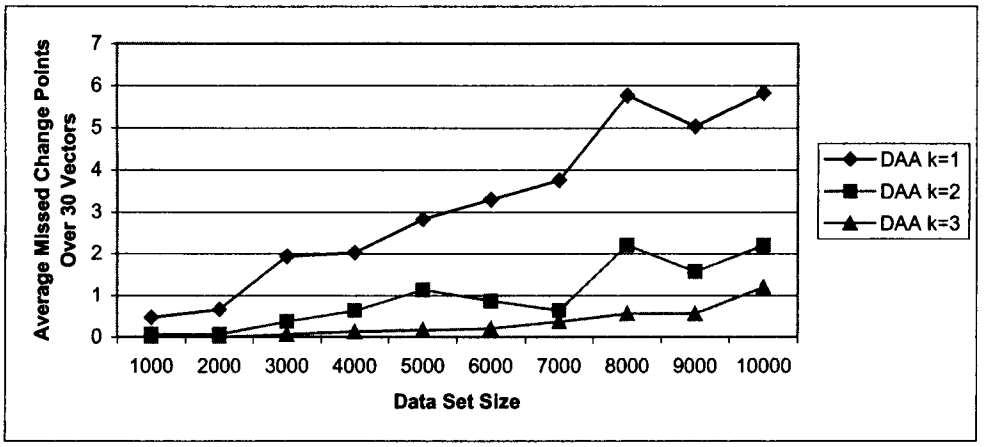


Figure 2.3 Average Number of Optimal Change Points Not Found by Each Algorithm

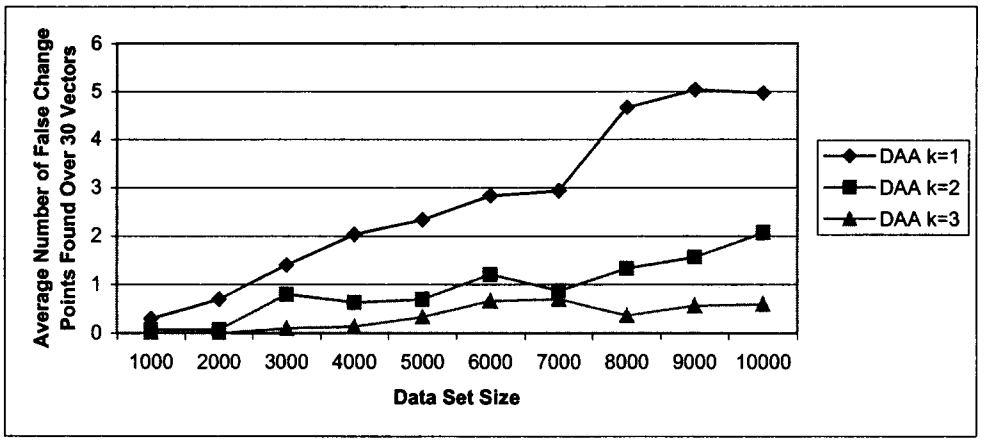


Figure 2.4 Average Number of False Change Points Found by Each Algorithm

Again, as the size of the data increases, performance decreases slightly. More change points occur in the large sets though, so relatively speaking, the algorithms give about the same quality of results for each data size. Again, improvement is seen with higher values of *k*.

Finally, the time required for each approximation is of interest. Each seems to perform fairly well, and higher values of *k* bring better results. The cost with respect to time of this increased computation must be addressed. Figure 2.5 shows average times for each algorithm, and it seems that, at least for these values of k, there is little difference between the versions of DAA.

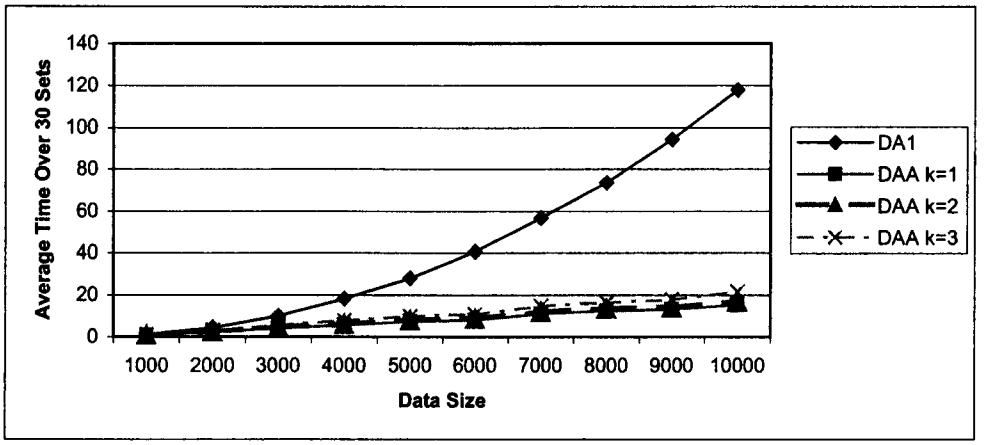


Figure 2.5 Average Times for Each Algorithm to Return a Solution

The rate of increase in time with the size of the data is far better for the approximations than DA1. The rates for the versions of DAA appear almost linear, but that is likely not the case. If the prior on the number of blocks mentioned in (1.3) is used when processing data with few blocks, the time performance will decrease slightly, since larger intervals with no significant fluctuations in data will be found. With the use of this prior the accuracy of all the approximations should increase.

Each version of the approximation algorithm gives good results here, both in terms of time and accuracy. As sets become more structured, and as the prior on the number of blocks is employed, some of the time advantages seen here might not be so great. With this however, the models returned by versions of DAA should become more accurate. Regardless, a fairly large decrease in the time required to find a solution should be achieved, with little loss of integrity for the model.

3 An Analysis of Algorithms for the Problem in Two Dimensions

For the problem in two dimensions, there is less information available to use in comparing algorithm performance. Algorithms guaranteed to find the optimal solution for an instance of the problem in two dimensions appear to be of exponential order. So the best partition is not at one's disposal to be used in evaluating the quality of approximations.

Problems also arise in gauging the true locations of the boundaries for a two-dimensional partition. Since the ideal boundary is not known, it may not be apparent which of two solutions returning similar results with differences occurring on the borders between blocks is a better model for the data. Changing the boundaries of a partition manually, and noting changes would allow one to evaluate the benefits of such changes, but with large sets or automated processing of data this seems unreasonable.

As these difficulties exist, the solutions returned by the various algorithms will be measured on the basis of two criteria: the objective function value, and the amount of time required for each algorithm to find a solution.

3.1 Algorithms for the Problem in Two Dimensions

Four algorithms were implemented to find solutions to the given problem with two-dimensional data. Each is a fairly common tool used in many combinatorial optimization problems. The defining factors of each will be highlighted here to illustrate the benefits and limitations of each algorithm.

One or both of two fundamental mechanisms for moving from one partition to the next are employed by all these methods. First, combining two connected groups of Voronoi cells to create a larger connected group will be referred to as a merge. Removing a single Voronoi cell from a larger connected group of cells will be a divide. This division seems limited, and it is, but it lends itself well to implementation since the additional computation and memory required is minimal. There a number of ways to divide up a block of Voronoi polygons. Choosing this method eliminates the problems encountered in allowing many or complex divisions, while adding enough flexibility to some algorithms to make up for poorly chosen merges. All the algorithms begin with the space B partitioned into Voronoi cells, no merges yet performed. The limitation that a cell may not be removed if doing so generates more than two connected blocks is also imposed.

Two of the algorithms are based on the principles of what are often referred to as greedy local search methods. The first of these is a merge-only algorithm. At every stage, the effect on the objective function is known for combining any two currently un-merged blocks. The effect of combining two blocks j and k is called a merge factor and is given in equation (5).

$$merge\,factor(j,k) = mf(j,k) = f(v_j + v_k, p_j + p_k) - [f(v_j, p_j) + f(v_k, p_k)] \qquad (5)$$

The two blocks with the greatest effect are merged, provided it yields an increase, and the process repeats. The algorithm terminates when no merge increases the objective function, or all cells have been merged to a single block. The merge-only greedy algorithm will be referred to as BM2.

The second of the greedy local searches is a merge/divide algorithm. It differs only in that at every stage the effect on the objective function of removing any cell contained in a larger block is also known. This is referred to as the divide factor, and the effect of removing cell j from cell k, or vice-versa is given in equation (6).

$$divide\,factor(j,k) = df(j,k) = [f(v_j, p_j) + f(v_k, p_k)] - f(v_j + v_k, p_j + p_k) \qquad (6)$$

The operation that corresponds to the largest merge or divide factor is performed if it increases the objective function. This repeats until no operation improves the objective function value. The merge/divide local search will be called BMD2. The remaining algorithms are based on the merge/divide local search. They differ in the decision process for operations and criteria to terminate.

Tabu search differs from local search methods only in that when a point is reached where no operation improves the objective function, or a local maximum, is reached, the algorithm continues. When no merge or divide yields an improvement on the objective function, the one that provides that smallest decrease is picked to be the next partition. That is, the operation that corresponds to the largest merge or divide factor, regardless of sign, is performed. Progressing in this manner, it is hoped that at some point a solution is encountered that improves on the one returned by the standard merge/divide local search.

As this algorithm searches the solution space some partition that is a local maximum is moved beyond. In this case, undoing the operation that was just performed, and returning to the local maximum, will certainly be beneficial. To avoid such a situation, a running list of the last t operations performed is retained. These operations are referred to as *tabu*, or invalid, and are retained in a *tabu list*. The length of the tabu list can be varied to change the output of the algorithm. This can help to fine tune the algorithm and find better results. Any operation that would be available in the best merge divide algorithm save those on the tabu list may be performed.

As local maximums will not terminate the algorithm, some other stop criterion is needed. Here, a limit on the number of iterations is imposed. This may be changed to accommodate the size of the data, time constraints, or allow longer runs. Since the algorithm does not necessarily stop at a maximum, the last partition found might not be the best one encountered during the search. To ensure that the best partition found is the one returned, a record of solutions is kept throughout the algorithm. Naturally, only those that improve the objective function are retained. When a partition is found that gives a higher value than all those found previously, it is recorded as the new best solution.

The final algorithm tried is simulated annealing, which is also similar to the merge/divide local search. For this algorithm, rather than choosing the solution with the largest merge factor and taking that as the next partition, one is picked at random. The operation prescribed is only proposed as a potential operation and is performed in accordance with an acceptance probability. The probability of performing non-beneficial operations is controlled by two parameters declared as the algorithm starts, and decreases as it progresses.

3.2 Performance Comparison for Algorithms in Two Dimensions

This comparison is intended to be only a preliminary investigation into the abilities of each algorithm to find near-optimal solutions to the given problem. On their own, the new algorithms may never prove to improve greatly on the results given by the local search methods. Future work may involve using an algorithm that combines local search and simulated annealing.

In the implementation of the algorithms, difficulty arose of deciding when a cell that is to be removed from a block corresponds to a cut-vertex of the dual graph for that block. In the case that it is a cut-vertex, the removal should be disallowed as it creates three disjoint blocks rather than only two, which is defined to be the limit of the algorithm. Methods examined to determine whether a vertex is a cut for a graph were taking the eigenvalues or finding the rank of the

Laplacian matrix of the graph. Even for moderately sized blocks this is a very expensive process in terms of time, so in any instance where this might occur only a small set of the block was considered. The rank of the Laplacian of the graph generated by the dual of the neighbors of the cell to be removed that lie within the block to which that cell belongs was found. Usually there are no more than five to ten vertices in such a graph. If this set was disconnected (rank < number of vertices – 1), then the removal was not allowed. In the future, other types of divisions might be explored to make the algorithms more flexible and to circumvent such problems as the one stated here.

Only one method for visualizing results is presented here. Each region, or block, of uniform intensity will be shaded with a different color. The largest region, here the background, will be shaded with white. Several other methods will likely be tried in the near future to better show changes in intensity in adjacent blocks.

To illustrate the performance of the algorithms, two sets of data with some known structure were generated. Low intensity background 'noise' was also imposed on the space to show regions of higher intensity being discovered within a larger set of points. The first of the two sets consists of three circular regions each of a uniform high-intensity distribution placed equal distances from a central point. This creates three distinct convex clusters within the background noise. The second of the two sets generates the same clusters, now placed fairly close together, each overlapping the others somewhat, to create a single non-convex region with several higher intensity regions within it. The two data sets are shown in figures 2.6 and 2.7.

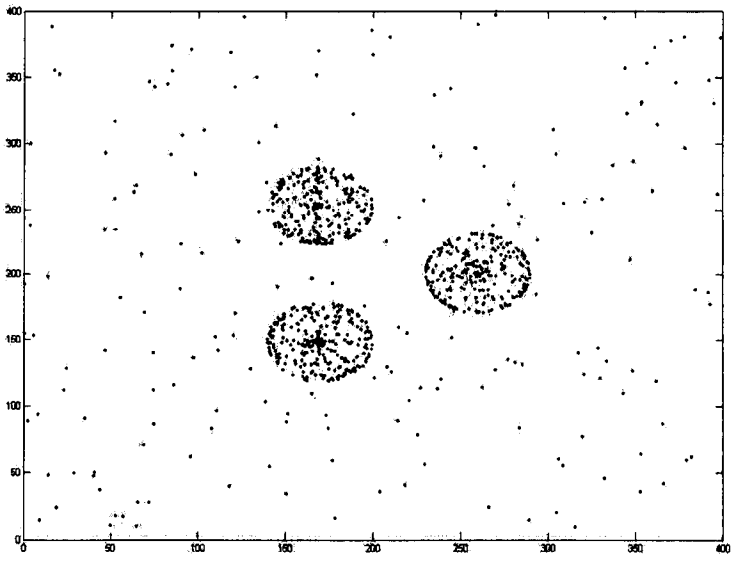


Figure 2.6 Data Set 1: Three Distinct Clusters

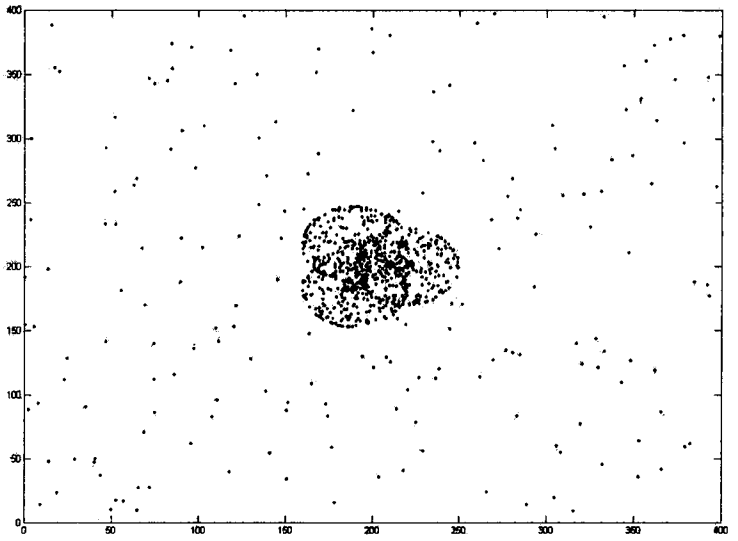


Figure 2.7 Data Set 2: One Large Cluster Containing Several Sub-Clusters

The merge-only and the merge/divide local search algorithms give the same result with the former finding it much more quickly. So the merge-only results will be shown to represent the performance of the local search algorithms. Table 2.1 summarizes the performance of all the algorithms tested on data set 1.

Table 2.1 Performance Comparisons for Algorithms on Data Set 1

	Time (seconds)	Objective Function Value	Blocks in the Model
Local Search	4.34	4358.9	50
Tabu Search	25.39	4358.9	51
Simulated Annealing	222.07	4443.8	51

The first two algorithms give nearly identical performance. Both found models with equal values for the objective function; the tabu search modeled the data with more blocks, however. Simulated annealing found a model with only one more block than the local search and a slightly worse objective function value. The times shown are expected for each of the algorithms. Both of the approximation methods, tabu search and simulated annealing give greater freedom in locating solutions, so with that comes a greater cost in time expended. The full power of both of these methods to find better solutions is not illustrated here since only a limited set of input parameters was tested for each. This at least shows these algorithms can return reasonable results, and with a more thorough search, could probably give consistent improvement over local searches.

The similarities between the models given by local search and tabu search are seen in the visualizations of the models given in figures 2.8 and 2.9. Also, the fairly significant difference in the structure of these models and that given by simulated annealing is seen when examining figure 2.10.

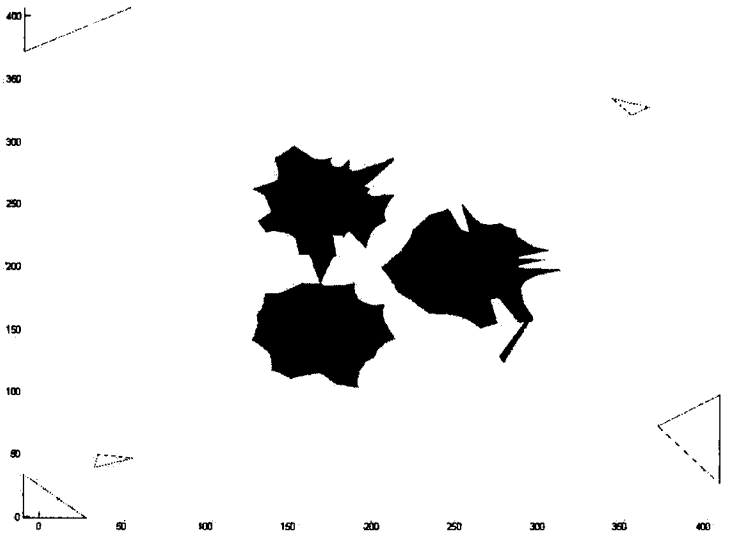


Figure 2.8 Local Search Results for Data Set 1

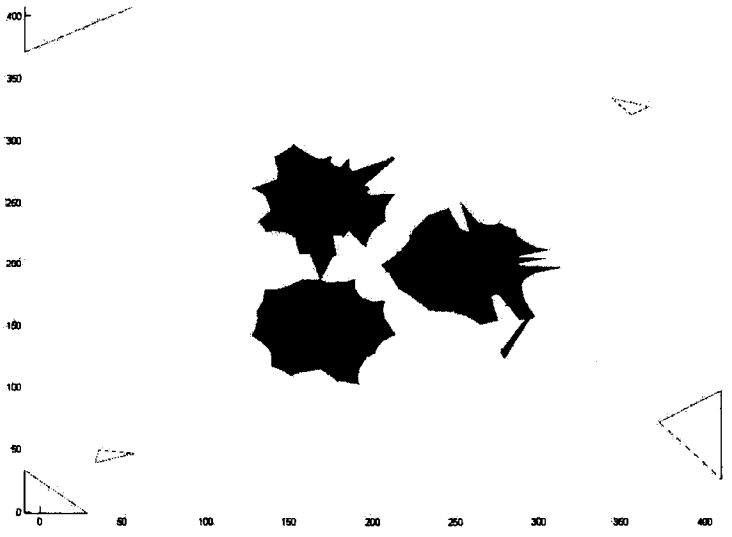


Figure 2.9 Tabu Search Results for Data Set 1

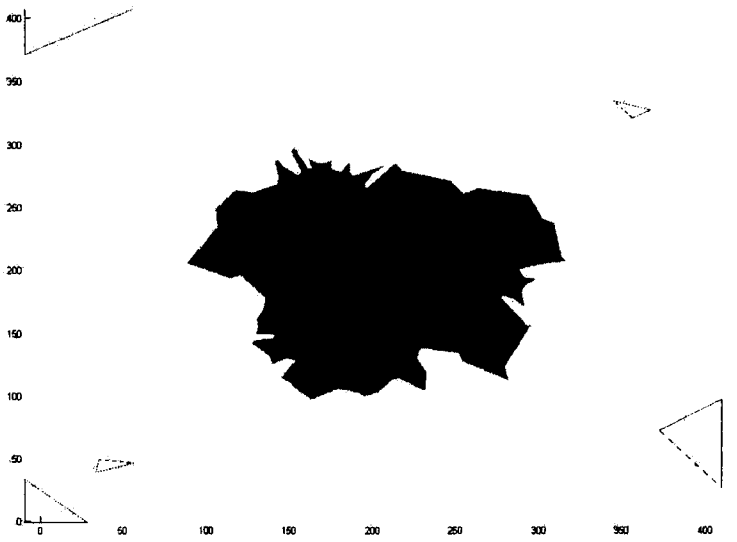


Figure 2.10 Simulated Annealing Results for Data Set 1

These visualizations were created by shading each block in a given model with a different color. No two blocks have the same color, but since the color scheme was chosen based on the intensity of the blocks, some adjacent blocks may have very similar shading. Other schemes utilizing a 3-D relief based on intensity have been investigated but not yet implemented. These would likely give a much clearer view of the block structure of a given set.

The results for each algorithm to process data set 2 are shown in table 2.2. In this instance, the parameters chosen as input for the tabu search (the same as were used for data set 1) gave a model that improved on the one given by local search. Again, simulated annealing did not perform well, but only the parameter set used on data set 1 was tried on this set.

Table 2.2 Performance Comparisons for Algorithms on Data Set 2

	Time (seconds)	Objective Function Value	Blocks in the Model
Local Search	4.25	3902.7	50
Tabu Search	24.67	3889.5	49
Simulated Annealing	255.05	3960.0	49

Figures 2.11 to 2.12 show the models visually. The difference between the local search and tabu search models is in the high intensity regions that lie on the interior of the large block. The model generated by simulated annealing again differs from the other two in a fairly obvious way.

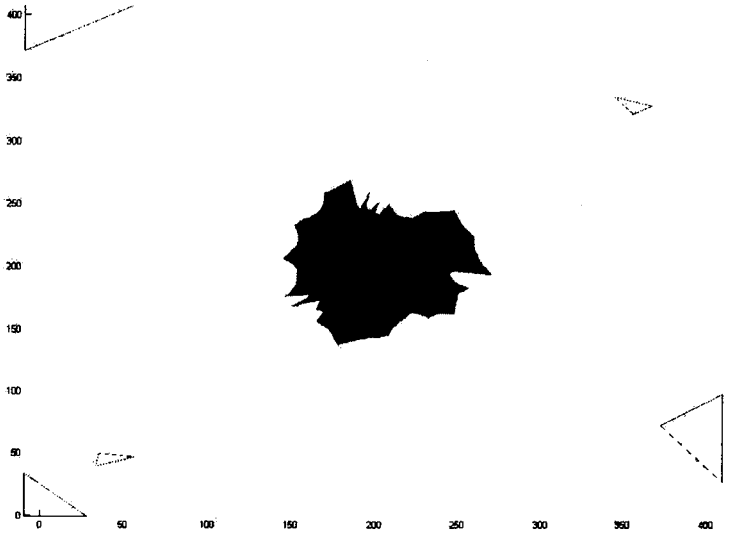


Figure 2.11 Local Search Results for Data Set 2

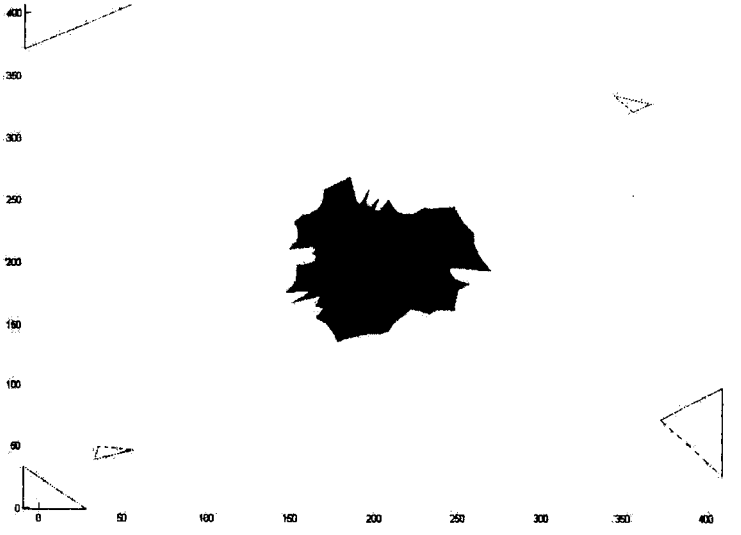


Figure 2.12 Tabu Search Results for Test Set 2

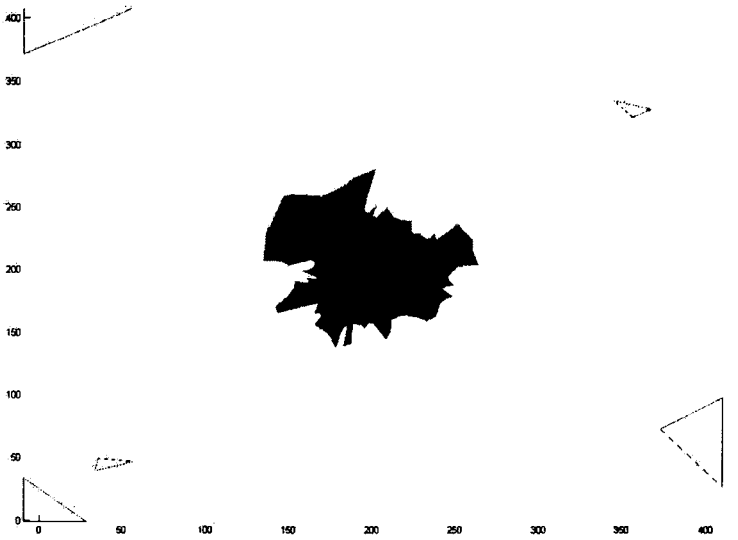


Figure 2.13 Simulated Annealing Results for Test Set 2

4 Conclusions

For the one-dimensional case great gains in improving the performance of this method will come in with a better understanding of the prior on the number of blocks in the model. Once a good estimate for this value has been made, an analysis similar to the one given here can be made on either real data, or simulations that mirror the type of structure expected in a particular application. From this a better approximation for k can be drawn.

The results for the algorithms to process two-dimensional data given here are somewhat early. A more complete search of the parameter space for both simulated annealing and tabu search will almost certainly improve upon what was shown. While each of the approximations shows promise, even a fairly deep search may not yield consistent improvement over local search for all data sets. Further work may involve utilizing two or more of these algorithms at the same time to find better solutions. Also, some changes may be made to the divide scheme so more operations are available at some stages.

Independent of the algorithms, the method seems very promising in effectively and accurately partitioning Poisson-distributed data. When the consideration is made of the complexity of the combinatorial optimization problem this method presents, it seems a bit less attractive because of the sizable investment of time for higher-dimensional sets. It is hoped that development of new, or improvement of existing search algorithms will help to set this concern to rest in the future.

A microtubule based control mechanism for perception-action behavior in a simulated eukaryotic cell.

Jeffrey Pfaffmann

Eukaryotic cells are capable of performing very complex information processing tasks, and can be thought of as a computational device capable of performing perception-action behavior. This behavior takes the form of the cell extracting information from the local environment, integrating the information relative to the current internal state, and producing an action to enhance the cell's fitness in the current environment. To facilitate this sort of information processing the cell would need to work in a coordinated fashion requiring a long-range signaling mechanism that can integrate information from across the cell quickly. Many short-range signaling mechanisms have been identified in the eukaryotic cell biology, but a long-range signaling mechanism has yet to be conclusively established. However, evidence indicates that the cytoskeleton could fill this long-range signaling role, specifically the microtubule network because of its organizational characteristics.

To explore this proposed signaling medium a learning model is used that combines a biologically motivated growth simulation with an abstract signaling mechanism to create an adaptive signaling medium. This signaling medium is molded by *adaptive self-stabilization* [1], which is essentially a feedback mechanism that translates network fitness into regulatory signals for modulating the growth dynamics. Ultimately a goal of this work is to harness the model's inherent oscillatory dynamics for controlling a biomimetic robot that captures the interdependent nature of the eukaryotic cell's various components.

The microtubule network is one of three components that make up the eukaryotic cell's cytoskeleton, playing the role of gross structural support and organizer of intracellular organelles. What sets the microtubule network apart from the other cytoskeletal components is its unique organization, which makes the network especially useful for efficient transport of proteins throughout the cell interior and potentially providing an effective long-range signaling medium. Microtubule networks consist of three basic components: microtubules, the Microtubule Organizing Center (MTOC), and Microtubule-Associated Proteins (MAPs). The microtubules themselves are large macromolecular tube-like structures composed of tubulin dimers that assemble by a growth mechanism called *dynamic instability* [2]. This growth mechanism produces a system where individual microtubules stochastically oscillate between assembly and disassembly, while the larger microtubule population maintains a stable net mass of assembled structure. The MTOC is a protein matrix around the cell's centrosome, which nucleates new microtubules while providing orientation to existing microtubules so that they radiate from the cell center to the outer cell periphery. MAPs are a large family of proteins that are capable of binding to the microtubules and allow the network to generate cell specific functionality. MAPs provide a variety of functions, including: stabilizing assembling microtubules, linking microtubules to other intracellular components, facilitating protein transport, and providing micro-muscle activity.

Several researchers have implicated the cytoskeleton as a mechanism for long-range signaling in the cell, specifically the microtubule network, because of its regular organization that reaches from the cell center to the periphery. Evidence for these interconnections have been shown to exist, whereby mechanical signals can be transmitted from the cell surface to the nucleus [3]. These interconnections start with proteins called integrins in the outer (plasma) membrane that are bound to the underlying cytoskeletal substructure, which in turn connects to the nuclear structure and allows a tug on the integrins to be transmitted to the cell nucleus. It was this interconnection and evidence for a long range signaling mechanism in the neuron that lead to the suggestion that the cytoskeleton, and specifically the microtubule network, is the neuron's "micro-nervous system". This resulted in a variety proposed signaling models in the literature [4], the majority of which are based on a type of vibratory dynamic.

Our learning model consists of three phases: growth, signal processing, and adaptive self-stabilization. During the growth phase, the microtubule network is allowed to develop in a three dimensional array of cubes that abstracts the intracellular environment and preserves the spatial arrangement found in the natural system (see Figure 3.1). The developed network is then reinterpreted by the signal-processing phase, which treats individual microtubules as strings of discrete oscillators that are interlinked by bound MAPs capable of actively modulating the signaling effects. Adaptive self-stabilization modifies the growth dynamics based on the fitness, which is derived from the network's signal processing capability. This fitness-based signal has the effect of stabilizing the microtubule network over time as the fitness value increases.

The behavior of the learning model is to begin with no microtubule network structure, which provides zero fitness since there is no network to support an information transform. Microtubule network structure will quickly begin to build up, although at this point the signaling pathways are incomplete and will provide a very low fitness. At some point a critical amount of network structure will be generated to support an effective set of signaling pathways, which will increase fitness and lead to a progressively more stable structure. This development of effective signaling pathways can be seen in experimental results presented in Figure 3.2.

The task given to the model for these results was to find the center point in a two-dimensional array from a random position. The learning model does this by taking the current location as input, integrating the information relative the current network dynamics, and producing an offset relative to the current array position for the next position. The process is then repeated with the new array position without resetting the internal network dynamics, providing a rough perception-action framework for the model to function within. In Figure 3.2 each rectangle shows the learning model's ability to perform the task for eight randomly generated points, with each arrow indicating direction and distance offset. At time points 10 through 54, little movement occurs beyond the initial random location generated and is due to the initial lack of developed network structure. However, as time progresses structure begins to develop that allows the model to begin producing useful effects and will eventually stabilize into more useful configurations (see time periods 211 and 562). It should be noted that the problem was partially encoded into the MAP functionality, allowing the learning model to achieve a "sense of direction" in the 2-dimensional array that can then be regulated by the correct placement of MAP on the microtubule network. Eventually this encoding will be replaced by an evolutionary

mechanism that evolves different MAPs types, eliminating the need for this kind of direct control over the model.

The learning model's signaling mechanism treats each microtubule as a string of simple discrete oscillators with neighbor-neighbor interactions. The reason for this representation is that a variety of possible vibratory dynamics can be represented in an abstract way; a second advantage of this implementation is that it provides a computationally simple simulation. MAPs are used in the signal processing mechanism to introduce active effects to the otherwise passive microtubule substrate, much the way MAPs interact with microtubules in the natural system. In the case of the learning model, MAPs: introduce signals into the microtubule network, exchange signals between neighboring microtubules, and extracts signals from the structure. As a result of this microtubule-MAP interaction MAPs produce a periodic triggering activity that mirrors the underlying microtubule network dynamics, producing behavior that is reminiscent of a spike train generated by a neuronal action potential.

To expand upon this simulation of information processing in the eukaryotic cell, the learning model will ultimately will be integrated with a biomimetic robot that abstracts eukaryotic cell functionality. The robot, called biot, consists of a sequence of twelve segments that are interconnected with both local and distant neighbor segments in a non-regular fashion to promote context sensitive interactions between the different segments. As a result of these complex interactions the current shape of biot cannot be calculated based on previous movements, instead a set of optical sensors are positioned throughout the device to provide a general feedback mechanism about the robot's shape. The resulting interactions between each segment produces movements similar to a biological system that when combined with the general feedback mechanism provides a very natural representation of a biological system.

The month spent at NASA-Ames Research Center focused on integrating the microtubule based learning model within the context of this perception-action framework of the eukaryotic cell. Additional work also examined various options of how to utilize the oscillatory dynamics inherent to the learning model so as to be an effective control mechanism for the biomimetic robot. This research avenue was followed for two reasons: first, the learning model functions well in a perception-action framework that would be required for manipulating the complex interactions needed to control the biot; second, the adaptive nature of the learning model would allow the biot to respond to changes in the local environment like encountering new terrain features. Additionally, by combining this learning model with the biot it would provide an abstraction for the exploration of the eukaryotic cell and how it coordinates the diverse intracellular activities to produce clear cohesive actions towards a goal that the cell perceives as necessary, usually resulting in a higher fitness and thus a better chance for continued survival.

Bibliography:

[1] M. Conrad. Emergent computation through self-assembly. *Nanobiology* 2:5—30, 1993.

[2] L. Wordeman and T.J. Mitchison. Dynamics of Microtubule Assembly *In Vivo*. In J.S. Hyams and C.W. Lloyd, editors, *Microtubules*, pages 287—301. Wiley-Liss, Inc., New York, 1994.

[3] A.J. Maniotis, C.S. Chen, and D.E. Ingber. Demonstration of mechanical connections between integrins, cytoskeletal filaments, and nucleoplasm that stabilize nuclear structure. Proceedings of the National Academy of Science USA (Cell Biology), 94:849-854, 1997.

[4] S. Hameroff, A. Nip, M. Porter, and J. Tuszynski Conduction pathways in microtubules, biological quantum computation, and consciousness. BioSystems, 64:149—168, 2002.

Figure 3.1- an example of the first 10 time steps of a developing microtubule population. Single black cubes represent new microtubules, while older microtubules are a black cube paired with a white rectangular cube. At time 0 the simulated cellspace is empty, at time 1 14 new microtubules are created, at time 2 10 of the 14 microtubules grow one array location, and so on.

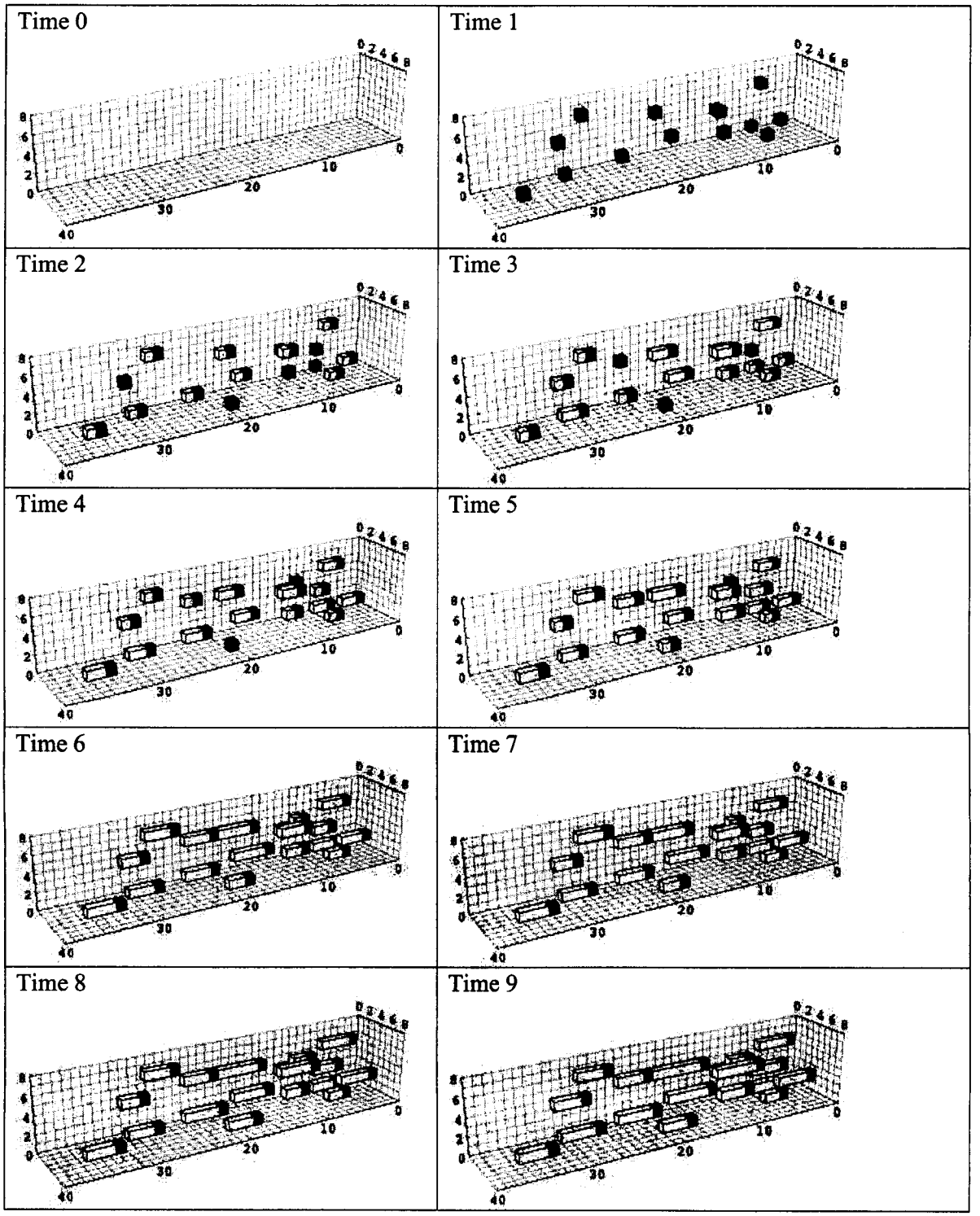
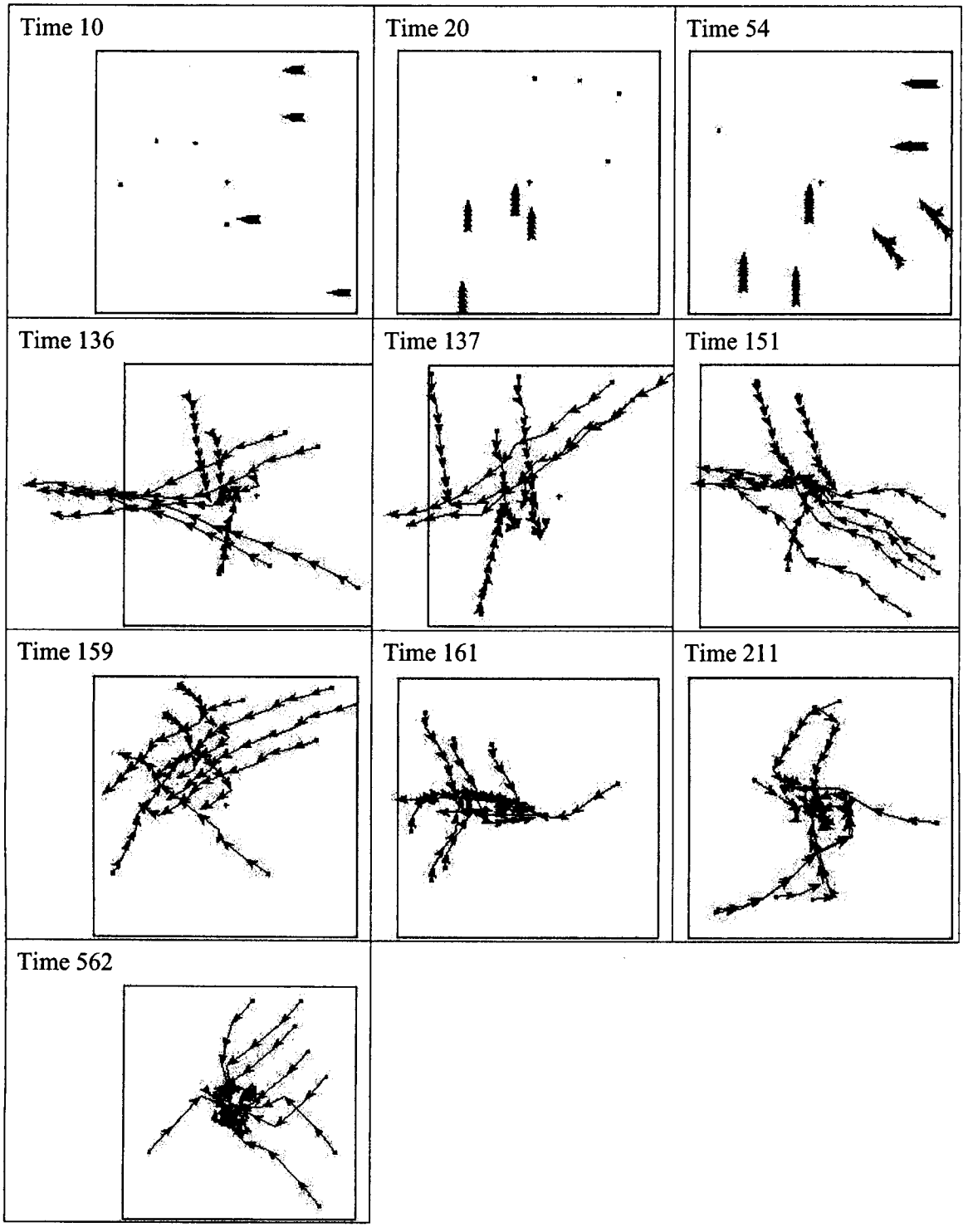


Figure 3.2 – experimental results where the learning model was given a random position in the 2-dimensional array and required to find the middle position. A black cross indicates the center and each start position is given as a black dot. The model's movement is given as a sequence of blue lines with an arrow to indicate direction and distanced moved. Fitness is determined based on the distance to the center, relative to the starting distance.



An inherited efficiencies model of non-genomic evolution

Natasha Li

This summer we examined one possibility for a mechanism driving the origin of life. One theory, the 'RNA world' hypothesis, describes RNA molecules acting as catalysts and mechanisms for information storage. It proposes information storage occurring prior to function. The other theory is one of non-genomic evolution, where function evolves prior to information storage. We assume peptide-bond forming proto-enzymes joined and broke apart amino acid chains. In this scenario, there is no information storage such as RNA, and the system evolves by increasing efficiency. Some of the peptides formed would have a tighter structure, therefore would be less prone to being hydrolyzed. Thus, the proteases would preferentially cut peptides with a less organized structure. These less organized peptides are assumed to have been less catalytically efficient, and therefore the preferential destruction of the less organized peptides would lead to an increase in total efficiency. Over time, this increase in total efficiency would indicate non-genomic evolution. This model, the inherited efficiencies model, was created by Michael New and Andrew Pohorille, and I was fortunate enough to work on updates of it this summer.

In essence, the model takes a protocell (a membrane-enclosed structure) and places random amino acids inside. Properties of the amino acids are sampled based on Gaussian distributions, and structure and efficiencies (for ligation and hydrolysis) are assigned. These amino acids can act as catalysts for the ligation or hydrolysis of other amino acids, or can participate as substrates in reactions. The previous model also included the concept of activation of polymers.

When I came into this project, Dr. New and Dr. Pohorille were examining a way of implementing the concept of similarity in the model. At that time, the only bias towards one polymer acting as a catalyst other polymers was length. The smaller the difference in length, the more probable a reaction. This is reasoned because it is unlikely a very short polymer would be able to hydrolyze or ligate two very long polymers. However, it has been discussed that a very long polymer could easily ligate two shorter polymers, and nothing was determined about this issue. Apart from length, there was nothing in the model keeping track of a particular polymer's characteristics. It was my job to research what the possible characteristic descriptors would be, and how they would interact in the context of our model.

Most of the initial research I did was examining Quantitative Structure-Activity Relationships (QSAR). These descriptors are used in biomolecular research to empirically determine characteristics about molecules, and predict molecular interactions based on these biological descriptors. There exist hundreds of QSAR descriptors, but most of them fall into several main categories of parameters: solubility, lipophilic, electronic, steric, constitutional, topological, geometrical, electrostatic, and quantum-chemical. Of particular interest to our research were parameters related to charge, dipole moment, polarity/hydrophobicity, steric effects, hydrophobic surface area, radius of gyration/flexibility, and number of hydrogen bonds (or other measures of internal stability). We were able to separate the important factors into two separate mathematical

groups: those descriptors which orientation dependant, and those which are not. For example, when two polymers are interacting, the interaction of the dipoles is extremely orientation dependant. At some orientations they might attract each other, and at others they would repel each other.

Therefore, for the first step, we wanted to include a term in the calculation of the reaction probabilities that would incorporate these similarity descriptors. If a catalyst and its substrates had different properties, it would decrease the likeliness of the reaction occurring. Initially, we would incorporate only descriptors which were not orientation dependant. Therefore, we examined descriptors such as net charge, polarity, and flexibility. Opposite charges would be more likely to attract polymers to each other. Polymers with similar hydrophobicities would be drawn together, and polymers with greater flexibility would have an easier time changing shape to accommodate the shapes of other polymers. Currently, we have these descriptors implemented as additive. We assume net charge is additive, as is an index of how connected/flexible a molecule is. The issue of polarity is debatable- since polarity is usually determined regionally, it is difficult to come up with a singular polarity index. If two molecules of similar hydrophobicities combine, they are likely to create a polymer with a longer nonpolar region, increasing the total hydrophobicity. However, it is possible that they would collapse the hydrophobic regions more effectively in the center and shield them, changing the overall polarity index. In any case, the implementation of other specific characteristic properties can be changed with little effect to the model.

When attempting to introduce a term for similarity, it became apparent that the old model could not incorporate the term easily. Previously, the total probability for a ligation between two polymers or the hydrolysis of a polymer was easily expanded from the particular probabilities of each reaction occurring. Previously, we calculated probabilities of ligation between a and b with catalyst c, from $\tilde{\pi}_{ab;c} \rightarrow \tilde{\pi}_{ab} \rightarrow \tilde{\Pi}_L \rightarrow \Pi_L$. In the original model, $\tilde{\pi}_{ab} = \frac{1}{N(N-1)} \sum_{\substack{c=1 \\ c \neq a,b}}^{N} \varepsilon_c^L = \frac{\varepsilon_{tot}^L - \varepsilon_a^L - \varepsilon_b^L}{N(N-1)}$. However, with the implementation of specificity, the model became: $\tilde{\pi}_{ab} = \frac{1}{N(N-1)} \sum_{\substack{c=1 \\ c \neq a,b}}^{N} \varepsilon_c^L \cdot f(\vec{w}_a, \vec{w}_b, \vec{w}_c)$, where $f(\vec{w}_a, \vec{w}_b, \vec{w}_c) = e^{-\alpha \left| \vec{w}_a + \vec{w}_b - \vec{w}_c \right|^2}$, and $\vec{w}_a, \vec{w}_b, \vec{w}_c$ are vectors of characteristic properties of each polymer. This probability was more difficult to enumerate than the previous one, and it was decided a new strategy would be needed in order to program specificity efficiently.

After reading Gibson and Bruck's article, "Efficient Exact Stochastic Simulation of Chemical Systems with Many Species and Many Channels" (1999), we decided to adopt their algorithm of the Next Reaction Method. In essence, we would write a system of possible chemical equations (in our example, ligations or hydrolyses of polymers), and calculate the propensities, or k's, for each reaction:

$A + B + C \xrightarrow{k_1} C + D$ (ligation with catylist C)

$A + C \xrightarrow{k_2} E + F + C$ (hydrolysis with catylist C)

etc...

Now, using these k's, for each reaction u, we can generate a putative time, τ_u, at which reaction u occurs. Then, we choose the reaction with the smallest τ_u and execute it.

When we decided to use this method, it was evident that a significant rewrite of the code was needed. Since I am more familiar with C programming that FORTRAN, and since the data structures we could program in C would make the information storage easier, we decided to recode the inherited efficiencies model in C.

Essentially, the program works like this:

There are three main data structures. One is a structure representing each monomer/polymer. These polymers are linked together in a doubly linked list formation. Each polymer structure contains information about the polymer such as efficiencies of ligation and hydrolysis, structure, length, activation state, and characteristic properties. Each polymer is also the head of two lists- one list represents the reactions it is a substrate in, but does not catalyze. This list is basically a linked list of structs which only contain pointers to reaction structures (explained next). In other words, this list of structures points to reactions that other polymers catalyze, which this particular polymer is a substrate in. These structs point to the real reaction structures, and do not contain any information about a reaction. The second list a polymer heads is the list of reaction structures that it catalyzes. For the purpose of the simulation, this was designed as a minimum priority heap. The heap is implemented in array form, where the children at position k are at 2k and 2k+1. The heap is sorted based on τ_u, using a version of heapsort. Each of these reaction structures contains pointers to the catalyst (i.e. the polymer which heads the heap) and the substrates. It also contains information about the reaction such as the reaction speed and τ_u.

The program executes by assembling the list of monomers and possible reaction lists for each monomer. Then, the reaction u with the lowest τ_u is searched for, and this reaction is executed next. The program follows the pointers from the reaction to the polymers it uses as substrates, and deletes all the pointers to their reaction heaps and reaction heaps. It then samples a new polymer from the characteristics of the old polymers, and then deletes the old polymer. In the case of a hydrolysis, the length of the first polymer is selected from a uniform distribution between 1 and n_a, where n_a is the length of the substrate. The second polymer is then assigned the difference between n_a and the first polymer. The efficiencies are selected from gaussian distributions, and characteristic properties are selected from uniform distributions the same way as length. The structure is calculated from efficiencies using one of three models (the same used in the previous simulation). In the case of a ligation, the length and characteristic properties are summed. Then, the program creates a reaction heap under the new polymer of the possible reaction it can catalyst, and then creates new reactions in the other polymers' reaction heaps containing this new polymer. Finally, the polymer is inserted, and the number of monomers is

checked. If the number of monomers has changed, the program increases or deletes the monomers until the correct number is reached.

There are many limitations of the program as of now. First, it doesn't completely run. It gets caught up in assembling the monomer list somewhere around the 50^{th} monomer. It was getting caught in the sampling function of tau, but now I think it may be a memory problem. Also, I have not yet programmed in anything about activation. There are global variables assigned and inputs read regarding activation parameters, but this is not implemented. Furthermore, while there are system outputs after every set number of reactions executed, there is no code to write these outputs into a file. Finally, I selected tau from an exponential distribution using the k's as the parameter. I used Michael New's code for sampling from an exponential distribution, and apparently the method used in the Next Reaction Method could refer to several ways of using the k's as a parameter. This should be investigated and clarified.

Astrobioinformatics Database Installation

Zhihui Xiao

Introduction

This summer I worked on the Astrobioinformatics project as an intern at NASA-Ames Research center. I really appreciated that NASA gave me this chance to have this wonderful experience.

My work focused on the database part. My main job is to install Stanford Mircoarray Database based on Oracle Server software. As we all know, the expansion of bioinformatics is expected to lead to an unlimited number of discoveries and also experiments. Thus we need a database which can store our huge amount of Microarray experiments. Most of these experiments are done on space which are too expensive to afford any lost or crash. SMD (Stanford Microarray Database) stores raw and normalized data from Microarray experiments, as well as their corresponding image files. In addition, SMD provides interfaces for data retrieval, analysis and visualization. Data is released to the public at the researcher's discretion or upon publication. So, SMD is one of the best approaches to achieve our goal.

SMD is making its full source code available free of charge to academic and non-profit institutions. It comes with some installation instructions and documentations, but it is still a very tough task to get SMD up and running. It requires Oracle Server, database administrating with Oracle experience, web server (Apache), system administrating, Perl and various modules, also some Perl and C program debugging. It was very challenging and fun!

Documentation

Here I want to document my SMD installation on Linux (Redhat 7.3) in five parts.

Part 1) Java Installation

Oralce requires both the JRE and JDK versions 1.18. I went to the Blackdown JDK/linux website to get the JDK-jdk118_v3-glibc-2.1.3.tar.bz2 and JRE-jre118_ve-glibc-2.1.3.tar.bz2. After uncompressing and untaring both of these files, I made the following symbolical links:

```
ln -s /usr/local/jdk118_v3/bin/java /bin/java
ln -s /usr/local/jre118_v3/bin/jre /bin/jre
ln -s /usr/local/jdk118_v3 /usr/local/java
ln -s /usr/local/jre118_v3 /usr/local/jre
```

Part 2) Oracle Installation

The main point to get Oracle installed is to read a bunch of documents and try to follow the correct instructions according to our own system. When I had questions, I usually went to Oracle online forum to get technical support. As follows is a short documentation for the key points.

1. unpacking downloaded oracle 9i installation files and burning Oracle9i CDs

zcat lnx_920_disk1.cpio.gz | cpio -idmv
zcat lnx_920_disk2.cpio.gz | cpio -idmv
zcat lnx_920_disk3.cpio.gz | cpio -idmv

or using CD:

mv lnx_920_disk1.cpio.gz lnx_920_disk1.cpio
cpio -idmv lnx_920_disk1.cpio

2. Kernel Parameter settings for linux
[root@oracle etc]# grep MemTotal /proc/meminfo
MemTotal: 1030480 kB

[root@oracle etc]# cat /proc/swaps

Filename	Type	Size	Used	Priority
/dev/hda2	partition	2096472	0	-1

[root@camelot data]# cd /proc/sys/kernel
[root@camelot kernel]# cat sem
250 32000 32 128
[root@camelot kernel]# echo 250 32000 100 128 > sem
echo 2147483648 > shmmax
[root@camelot kernel]# cat shmmax
2147483648
[root@camelot kernel]# cat shmmni
4096
[root@camelot kernel]# cat shmall
2097152

3. Create Oracle user accounts
groupadd dba
groupadd oinstall
useradd -g oinstall -G dba oracle
passwd *****
mkdir /u01
chown oracle.dba /u01
chmod 775 /u01

4. Set Oracle Environments (as oracle)

.bash_profile

```
export ORACLE_BASE=/u01/app/oracle
export ORACLE_HOME=/u01/app/oracle/product/9.2.0
export ORACLE_SID=oral
export ORACLE_TERM=xterm
export NLS_LANG=AMERICAN;
export ORA_NLS33=$ORACLE_HOME/ocommon/nls/admin/data
LD_LIBRARY_PATH=$ORACLE_HOME/lib:/lib:/usr/lib
LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/usr/local/lib
export LD_LIBRARY_PATH

export PATH=$PATH:$ORACLE_HOME/bin


CLASSPATH=$ORACLE_HOME/JRE:$ORACLE_HOME/jlib:$ORACLE_HOME/rdbms/jlib
CLASSPATH=$CLASSPATH:$ORACLE_HOME/network/jlib
```

5. logout and login again as oracle

```
[root@oracle root]# chown oracle.oinstall Disk1 -R
[root@oracle root]# chown oracle.oinstall Disk2 -R
[root@oracle root]# chown oracle.oinstall Disk3 -R

mv Disk1 /home/oracle/oracle_install/
mv Disk2 /home/oracle/oracle_install/
mv Disk3 /home/oracle/oracle_install/
```

Part 3) LIBS Installation

SMD and Perl both require Linux libraries to be compiled and installed, such as libjpg, libfreetype, libxpm, zlib, and png. I downloaded a patched version of libgd to support GIF files. Also I downloaded libnetpbm verion 9.23. The documentation is omitted here.

Part 4) Perl and Perl Modules Installation

Perl comes installed with Linux. SMD also requires a few modules, such as CGI, DBI, DBD-Oralce, GD, Storable and LWP. The detailed documentation is also omitted.

Part 5) SMD Installation

It is a very long and hard work. During the long installation of SMD, I communicated a lot with SMD staff at Stanford and took active part in the Stanford Microarray Forum. Both of these helped me a lot.

Here I gave some introduction of SMD. SMD is composed of both Oracle database portion and web application portion. Its Oracle database portion includes SQL scripts to create a SMD database instance, create database table spaces (the size, segmentation, etc are optimized for SMD), create tables, constraints, and triggers, and create database roles and users. Its web portion provides PERL scripts to authenticate SMD user, provide users with different privileges, give them different views of the SMD data according to their privileges, load external annotations and keep them up to date, add/edit/delete/search organisms, prints, experiments, and some other functionalities and tools.

The connection to underlying database in SMD is handled through db-auth module, which provides a way to avoid hardcode Oracle user/password in the PERL scripts, so that it avoids the risk that someone can get to know database user/password by looking at the script source code.

Two main database users created by SMD is tableowner and webowner. tableowner is the owner of all the SMD tables, and has the privileges to create, read, edit and delete them; while webowner can only read data in SMD tables. An SMD user has various attributes that will affect what privileges they have, to define what experiments they can see, whether they can edit them or not, and what other database actions they can perform through web interface.

Here I don't include the documentation for the SMD. All the documentations could be found at Camelot.farm.network under /smd/myDocs.

Comments and Acknowledgements

During the internship, I also get to know a lot of other intern students, with whom I share the same interest in bioinformatics.

I want to thank Andrew for giving me this chance to have such a wonderful experience. Also thanks for the help from Andrew, Karl and Mike, both on technical and non-technical issues. It's so nice of them to find as many chances as possible to introduce us the development of biology, space-technology, and computer science. I really appreciate the seminar series, which help me to understand more about the mars, biology etc. During these seminars, we get the internal and advanced knowledge from scientists and experts, and also share our ideas and opinions. Also thank them for organizing so many interesting activities to bring us a really wonderful summer time.

In a word, I believe this internship experience is mutually benefited. I hope more students will have this chance in the future!

Improving the Cosmos molecular dynamics program using multi-timestepping techniques

Galen Reeves

Introduction

Multiple-timestepping techniques have the ability to improve the speed of a molecular dynamics simulations. These techniques sample the expensive and slowly changing long-range forces less frequently than the short-range forces. Tuckerman and Berne have developed an integrator called reversible reference system referencer algorithm (rRESPA) [3] based on the Trotter expansion of the Liouville Propagator. The (rRESPA) algorithm separates the long and short range forces in a way that is reversible, and therefore conserves energy and has some degree of stability.

In this report, the Trotter expansion used in (rRESPA)is has been implemented into the Cosmos molecular dynamics program [2] using the forces from the particle mesh Ewald (PME) as the long range forces. The method created, Cosmos-Trotter, is tested on a water-hexane system and a protein system.

Improved speed in molecular dynamics simulations will allow for the study of systems for longer time intervals. Many biological processes occur on the order of milliseconds and the applications of the md program increase with its simulation time.

Implementation

The Trotter expansion fits in well with the basic structure of Cosmos and is able to use the velocity Verlet routine to iterate the system. To run cosmos-Trotter, a short time step, δt, and a long time step, Δt, need to specified. The value of δt is given by the original time step supplied in regular Cosmos, and Δt is given by the parameter *ntrott*, where $ntrott = \Delta t/\delta t$.

The forces are split into short-range and long-range components. The short-range forces are determined from the straightforward calculations of all the atoms up to a given cutoff length. The long-range forces are calculated using the particle mesh Ewald (PME) sum.

Currently, there are several restrictions on the input parameters. On the steps where the PME is not run, the total energy of the system cannot be calculated. Therefore, energy statistics should be taken only on Δt steps. Also, to assure that a run finishes correctly, the total number of steps in the run be a multiple of *ntrott*.

The Algorithm. The program uses the velocity Verlet algorithm to propigate each step. A regular velocity Verlet step on a system takes $x(t) \rightarrow x(t+\delta t)$ and $v(t) \rightarrow$

$v(t+\delta t)$.

$$v \leftarrow v + F\frac{\delta t}{2m}$$
$$x \leftarrow x + v\,\delta t$$
calculate F
$$v \leftarrow v + F\frac{\delta t}{2m}$$

In the Cosmos implementation of velocity Verlet, the order of the algorithm is changed slightly to put the force calculations at the beginning of the step. The explicit implementation consists of the following:

calculate F
$$v(t) \leftarrow v(t - \frac{\delta t}{2}) + F(t)\frac{\delta t}{2m}$$
perform velocity rattle
$$v(t + \frac{\delta t}{2}) \leftarrow v(t) + F(t)\frac{\delta t}{2m}$$
$$x(t+\delta t) \leftarrow x(t) + v(t + \frac{\delta t}{2})\frac{\delta t}{2m}$$
perform position and velocity shake

Throughout a single step,$F(t)$ has the same value. A step still takes $x(t) \rightarrow x(t+\delta t)$. However, the velocities of the system are taken from $v(t - \frac{\delta t}{2}) \rightarrow v(t + \frac{\delta t}{2})$. When the total energy for a given step is calculated, $v(t)$ from the most recent step, and $x(t)$ from the preceding step are used to describe the system at at time t. In addition, this implementation requires that $v(t-\frac{\delta t}{2})$ and $x(t)$ are somehow obtained for the beginning of a run. Therefore, runs usually start and end with $v(t - \frac{\delta t}{2})$ and $x(t)$.

The basic trotter algorithm breaks the contributions from short-range, F_{short}, and long-range, F_{long} forces. With $ntrott = \Delta t/\delta t$, the algorithm is:

$$v \leftarrow v + F_{long}\frac{\Delta t}{2m}$$
do $i = 1, ntrott$

$v \leftarrow v + F_{short}\frac{\delta t}{2m}$

$x \leftarrow x + v\,\delta t$

calculate F_{short}

$v \leftarrow v + F_{short}\frac{\delta t}{2m}$

end do

calculate F_{long}

$v \leftarrow v + F_{long}\frac{\Delta t}{2m}$

This implementation takes the system from $x(t) \rightarrow x(t + \Delta t)$ and $v(t) \rightarrow v(t + \Delta t)$. However, to integrate the algorithm into the structure of the cosmos code, it is necessary to arrange the step so that the force calculations take place at the end of a step rather than in the middle. In this description of the cosomos implementation, F_{long} is replaced by F_{pme}.

do $i = 1, ntrott$

clear F_{total}

if $(i = 1)$, then

calculate F_{pme}

$F_{total} = ntrott * pme$

endif

calculate F_{short}

$F_{total} = F_{total} + F_{short}$

$v \leftarrow v + F_{total}\frac{\delta t}{2m}$

perform velocity rattle

$v \leftarrow v + F_{total}\frac{\delta t}{2m}$

$x \leftarrow x + v\,\delta t$

perform position and velocity shake

end do

This arrangement breaks the algorithm into a part that sets up the the forces, and a part that propagates the system using velocity Verlet. The positions of the system

go from $x(t) \rightarrow x(t+\Delta t)$. However, the velocities between steps are not $v(t-\frac{\delta t}{2})$ because they are based on the forces $(ntrott\ F_{pme})+(F_{short})$ and do not correspond to an actual time. I will call the velocity at this time $v(\xi)$. After the first propagation step of the algorithm the velocities are moved to $v(t)$. Therefore it is easy to show how the velocities can be converted from $v(t-\frac{\delta t}{2})$, the form in which the velocities are stored, to $v(\xi)$, the form appropriate for the *trotter* algorithm. Here, F_{Verlet} equals the normal Verlet forces.

$$
\begin{aligned}
v(\xi) &= v(t) - (ntrott\ F_{pme}\frac{\delta t}{2m} + F_{short}\frac{\delta t}{2m}) \\
v(\xi) &= v(t) - F_{Verlet}\frac{\delta t}{2m} - (ntrott-1)\ F_{pme}\frac{\delta t}{2m} \\
v(\xi) &= v(t-\frac{\delta t}{2}) - (ntrott-1)\ F_{pme}\frac{\delta t}{2m}
\end{aligned}
$$

An additional startup routine is required to convert the velocities from $v(t-\frac{\delta t}{2}) \rightarrow v(\xi)$ by subtracting $[(ntrott-1)\ F_{pme}\frac{\delta t}{2m}]$ from $v(t)$. Also, a shutdown routine reverts the velocities to the standard from by adding $[(ntrott-1)\ F_{pme}\frac{\delta t}{2m}]$ to $v(\xi)$.

The Code. Few code changes are necessary to make the algorithm work with cosmos. One additional parameter, *ntrott* needs to be supplied to the program through the input file. The routine **mdfin** , which adjusts the velocities at the end of a run, is the only new routine in the program. All other changes were modifications to existing routines. Below is a brief description of their additional functions.

getopt reads the value for *ntrott* from the input file.

iter8 keeps track of the step number. For a step size Δt nrgfrc is passed the value of ntrott. For a step size δt nrgfrc is passed the number 1.

nrgfrc calls pme only if ntrott is greater than zero.

pme calculates pme forces. If *ntrott* equals zero, then the pme forces from the previous force call replace the total forces. If *ntrott* is greater than zero, then the pme forces are calculated, multiplied by ntrott, and added to the total forces.

main runs both mdstart and mdfin.

mdstart performs normal operations and, if *ntrott* is greater than one, subtracts $(ntrott-1)$ times the velocities from the pme, from all the velocities.

mdfin is called at the end of a run. If *ntrott* is greater then one, all velocities are adjusted by adding $(ntrott-1)$ times the velocities from the pme.

Method	δt	Δt	drift(wat-hex)	drift(protein)
Verlet	1	...	-0.09	-0.2
	2	...	-0.08	-1.18
	3	...	-0.33	-3.34
	4	...	-0.56	-6.04
	5	...	-0.19	18.6
	6	...	1.02	
	7	...	18.6	
Trotter	1	2	-0.07	0.15
	1	3	-0.09	0.08
	1	4	-0.07	0.26
	1	5	-0.07	0.45
	1	6	-0.01	0.45
	1	7	-0.06	0.60
	1	8	0.05	1.21
	1	9	0.00	2.11
	1	10	1.44	9.36

Table 1: Comparison of energy drift per step in wat-hex and protein systems. The short-range time step, δt, and the long-range time step, Δt, are both fs. The energy drift, is given in units of $(kcal\, mol^{-1}\, ps^{-1})$.

Analysis

The performance of the cosmos-Trotter was tested with two systems: a water-hexane system(wat-hex)consisting of 6382 water molecules and 450 hexane molecules, and a water-protein system, (protein), consisting of 5020 water molecules and 5808 protein atoms. In all tests, the wat-hex system was run for 5 ps at 300K, and the protein system was run for 2 ps at 310K. The systems where tested with different different step sizes, Δt and δt. The total energy of the system, potential plus kinetic, is analyzed to see how stable it is throughout the run. A drift in energy indicates that the system is not conserving energy and implies that the trajectories can not be accurate. If the system is run a for a long time with an energy drift the temperature of the system starts to increase. This requires the velocities to be periodically sampled from a distribution to reset the temperature. This destroys the reversibility of the run.

Table 1 shows the energy drifts associated with the two systems. Figure 1. shows a comparison between the Verlet and Trotter methods for the wat-hex system. The Verlet can be expanded to a step size of about 2 fs but breaks down at longer time steps. The Trotter is stable up to 9 fs. In the protein tests, Fig. 2, the differences between the methods are more pronounced. Because the protein system contains more small molecular bonds which vibrate at a high frequency (on the order of

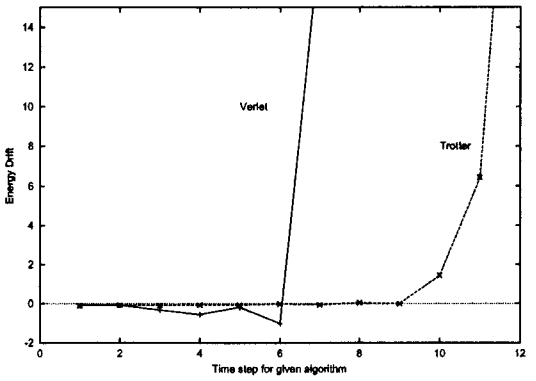


Figure 1: (wat-hex) Energy drift for step parameter. The step paramater is δt (fs) for Verlet, and Δt (fs) for Trotter. The energy drift is given in $\Delta E/ps$.

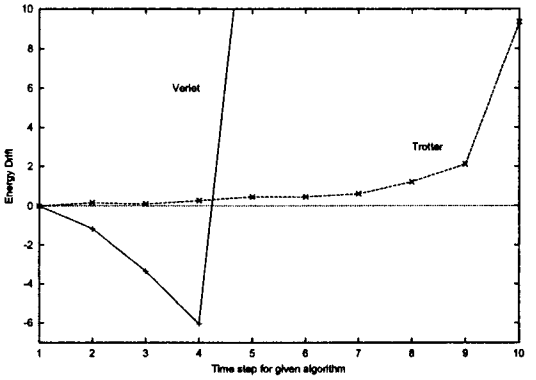


Figure 2: (protein) Energy drift for step parameter. The step paramater is δt (fs) for Verlet, and Δt (fs) for Trotter. The energy drift is given in $\Delta E/ps$

fs) the Verlet method breaks down immediately. The Trotter method continues to calculate the bond forces at 1 fs and remains quite stable for up to around $\Delta t = 7$ fs.

The Trotter method produced a increase in speed for both system. Fig. 3 shows the cpu time spent per step. These times are only approximations because that the program was not run with optimal speed settings and the conditions may vary. However, a decrease in time is shown. The wat-hex sytem goes from around 4.25 to around 2.5 seconds/step while maintaining stability. The protein system goes from around 5.9 to 4.2 seconds/step maintaining stability.

Additional Results

The Trotter algorithm is limited in part by the periodic oscillations that result from

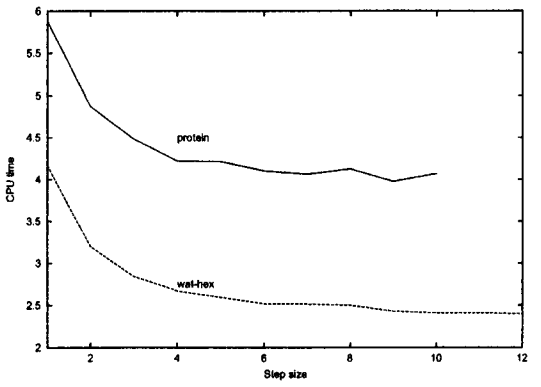


Figure 3: CPU time (seconds) per step. Note that the wat-hex system is stable to around $\Delta t = 9$ and the protein system is stable to around $\Delta t = 7$.

introducing "large" forces every Δt steps. If the longer time step is a multiple of the phase of the molecular bond vibrations it will start to drive vibrations and the simulation will fail.

One attempt to reduce this effect, an extrapolation/correction was tried on the system. [1] Rather than include $ntrott\,pme$ forces every *ntrott* steps, the pme forces were added every step, but only updated every *ntrott* steps. Since the system has a normal sized force each step, it was hoped that the outer time step could be increased without amplifying the bond occlusions. Additionally, a correction was put on the velocities every step to attempt to reduce the error introduced. This was a non-reversible system and the results were horrible. Even with very small step sizes the energy took off drastically.

An additional parameter that has not been tested is the cutoff length for the short-range forces. A cutoff length half the length of the simulation box has been suggested as a good setting[1] and adjusting the cutoff length for a particular system could improve the trotter results.

Furthermore, T. Schlick has suggested that stability can further be increased with the use of a position Verlet iterator instead of the velocity Verlet iterator.

Conclusions

This report has shown that is is possible to implement the Trotter algorithm into Cosmos with a decrease in simulation time. For both the water-hexane and the water-protein system, the PME forces could be calculated less frequently than the short range forces. This allows for longer, and more accurate simulations. When running Cosmos, care must be taken to record energies only from steps with full force calculations, and to ensure that a run ends on a full force calculation.

References

[1] E. Barth and T. Schlick. II. Extrapolation versus impulse in multiple-timestepping schemes: Linear analysis and applications to newtonian and langevin dynamics. *J. Chem. Phys.*, 109:1633–1642, 1998.

[2] B. Owenson, A. Pohorille, M.A. Wilson, M.H. New, and E. Darve. COSMOS — *A software package for COmputer Simulations of MOlecular Systems.* NASA — Ames Research Center, Moffett Field, CA 94035–1000, 1987.

[3] M. Tuckerman, B. J. Berne, and G. J. Martyna. Reversible multiple time scale molecular dynamics. *J. Chem. Phys.*, 97(3):1990–2001, 1992.