01 Aug 2008

Maximum Likelihood Methods in Biology Revisited with Tools of Computational Intelligence

John E. Seiffertt IV
Missouri University of Science and Technology, jes0b4@mst.edu

Andrew Vanbrunt

Donald C. Wunsch
Missouri University of Science and Technology, dwunsch@mst.edu

30th Annual International IEEE EMBS Conference
Vancouver, British Columbia, Canada, August 20-24, 2008

Maximum Likelihood Methods in Biology Revisited with Tools of Computational Intelligence

John Seiffertt, Andrew Vanbrunt, and Donald C Wunsch II, *Fellow, IEEE*

Abstract— **We investigate the problem of identification of genes correlated with the occurrence of diseases in a given population. The classical method of parametric linkage analysis is combined with newer tools and results are achieved on a model problem. This traditional method has advantages over non-parametric methods, but these advantages have been difficult to realize due to their high computational cost. We study a class of Evolutionary Algorithms from the Computational Intelligence literature which are designed to cut such costs considerably for optimization problems. We outline the details of this algorithm, called Particle Swarm Optimization, and present all the equations and parameter values we used to accomplish our optimization. We view this study as a launching point for a wider investigation into the leveraging of computational intelligence tools in the study of complex biological systems.**

I. INTRODUCTION

PARAMETRIC linkage analysis is a traditional tool employed by geneticists to discover the location of genes which contribute to diseases. The major advantage of this approach is a significant increase in statistical power when compared to non-parametric models. This particular tool has fallen out of favor in recent years due to the fact that in order to adequately model a complex biological system using the requirements of parameterization one must accept a highly computationally expensive algorithm. If models can be formulated that allow for looser restrictions and fewer assumptions no matter how many parameters may be involved in the model, then maximum likelihood methods may be employed with a high degree of success if computed using a feasible algorithm. Using methods that are known in the literature but which are just now being applied to complex biological problems, we show the utility of these parametric models. It is typical to use expectation maximization as the optimality criterion in various models of population genetics. Such uses range from haplotyping algorithms to modeling LD to non-parametric linkage models. In this paper, we show that the tools of computational intelligence can be brought to bear on this difficult biological problem and help to overcome the observed weaknesses associated with computational expense.

Population-based optimization methods are a topic of much current research effort. These algorithms seek to provide non-traditional approaches to standard optimization frameworks. In particular we investigate Particle Swarm Optimization (PSO) [3] in conjunction with the parametric linkage analysis problem. Introduced by Kennedy and Eberhardt and itself drawing upon principles observed in biological systems (primarily those of birds in flight) it has found application in a wide range of industrial domains. Properly situated as a specialization of the computational intelligence paradigm of Evolutionary Algorithms [1], we believe that PSO is ideally positioned to spearhead our investigations into applying a new series of computational tools to classical problems in biology and genetics.

Sections II and III detail our biological model, presuming familiarity with the basics of linkage analysis. Section IV gives an overview of our computational approach as well as explicit equations and parameters that we used, and section V concludes with remarks on advancing this line of research.

II. GENETIC SIMULATION MODEL

In our parameterization we assume a single unknown disease locus with alleles $D$and d. We also assume a single marker locus with an arbitrary (however, for simulation purposes, defined and fixed) number of alleles. We assume a normally distributed quantitative trait that is governed by the status of the disease locus. for the trait we assume three independent means μ_{DD}, μ_{Dd}, and μ_{DD} and a common variance σ^2. We further require that the disease locus be genetically linked to the marker locus. Additional model parameters used for simulation include a minor allele frequency q of the disease locus and a genetic distance between the marker and the unobserved disease gene given by θ such that $0 < \theta < .50$. When performing linkage this parameter θ becomes the most interesting because it indicates the proximity of the unobserved disease locus to the observed marker locus.

Nuclear family data was generated using the above parameters which could be fixed beforehand but blinded to

Manuscript received April 2nd , 2008. This work was supported in part by the Mary K Finley Endowment at the Missouri University of Science and Technology and in part by the National Science Foundation.

John Seiffertt is with the Department of Electrical and Computer Engineering, Missouri University of Science and Technology, Rolla, MO 65401 USA (e-mail: jes0b4@ mst.edu).

Andrew Vanbrunt is with the Division of Statistical Genomics, Center for Genome Sciences, Washington University School of Medicine, St Louis, MO 63108 USA. (e-mail: avanbrun@wustl.edu).

Donald C Wunsch, II is director of the Applied Computational Intelligence Laboratory, Department of Electrical and Computer Engineering, Missouri University of Science and Technology, Rolla, MO 65401 (e-mail: wunsch@ieee.org).

978-1-4244-1815-2/08/$25.00 ©2008 IEEE.

the MLE/PSO deconvolution program. Additionally, the number of families and the number of kids in each family could be easily controlled.

To simulate the data as we have described above, the basic process was to fix the components of the model (μ_{DD}, μ_{Dd}, μ_{DD}, σ^2, q , and θ), the desired number of families, the number of kids per family to generate, and the number of alleles of the observed marker locus and their frequencies, and to use q to simulate disease genotypes for mother and father. From the determined marker allele frequencies we can simulate marker genotypes. This must be done in an ordered fashion so initially the father's or mother's genotype has a given disease gene allele on the same chromosome as a given marker allele. These can be generated randomly but must be tracked in haplotype fashion.

A father's genotype consists of two haplotypes $\mathcal{D}_1 - \mathcal{M}_1||\mathcal{D}_2 - \mathcal{M}_2$. However, we are only able to witness genotypes and the data consists of marker allele genotypes as the quantitative trait. Therefore, the unobserved biological state is of the form $(\mathcal{D}_{12}\mathcal{M}_{12})$ without the observer knowing which $\mathcal{D}$ goes with which $\mathcal{M}$. For simulation purposes, this hidden state must be recorded to accurately make use of the recombination frequency θ and to generate disease-allele/marker-allele haplotypes for the parents of the nuclear families. Then a single parental haplotype is dropped for each child. This process is carried out for each parent, dropping one of their chromosomes to each child. Thus, the probability of no recombinations occurring in a family of n kids is $(1-\theta)^{2n}$, the probability of exactly one recombination is $\theta(1-\theta)^{2n-1}$, and so on.

Once each child has an intact or recombined haplotype from each parent, we then generate the quantitative trait/phenotype. These traits are drawn from a distribution dependent on the disease genotype: DD draws from $N(\mu_{DD},\sigma^2)$, Dd or dD draw from $N(\mu_{Dd},\sigma^2)$, and dd draws from $N(\mu_{dd},\sigma^2)$. Of course, when generating the input for the MLE/PSO solver, one would simply output the marker genotypes and familiar relationships.

III. Maximum Likelihood Estimation Method

One can accurately solve for the best estimates of the six model parameters given the data by setting up a likelihood equation. This translates to the idea of "what is the probability that we see the data we actually observe given that the parameters assume a particular set of values?" Or, $P(Data|\phi)$ where ϕ is a particular value in $\mathbb{R}^6$describing the values of the three phenotypic means and their common variance, the minor allele frequency, and the genetic distance or recombination fraction. To solve such a likelihood equation analytically would be feasible only in the simplest of models. However, it is relatively straightforward to apply Elston-Stewart's algorithm to re-write this likelihood in a manageable form that can readily be converted into computer code for simulation purposes. Matters simplify if we take a log likelihood to maximize.

For each nuclear family the likelihood of that family can be written as a product of each individual's phenotype given their disease genotype and marker alleles over all possible unknown disease genotypes. Thus,

$$L_{father}(phenotype) = \sum_{(DD,Dd,dD,dd)} P(i,m|phenotype).$$

Since the marker genotypes are equally likely, we can factor them out when maximizing the family likelihood.

With the frequency of D being q and the other aforementioned parameters, this sum takes the form

$$\sum = \frac{q^2}{(2\pi\sigma^2)^{1/2}} e^{-\frac{1}{2}\left(\frac{P-\mu_{DD}}{\sigma}\right)^2} + \frac{2q(1-q)}{(2\pi\sigma^2)^{1/2}} e^{-\frac{1}{2}\left(\frac{P-\mu_{Dd}}{\sigma}\right)^2} + \frac{(1-q)^2}{(2\pi\sigma^2)^{1/2}} e^{-\frac{1}{2}\left(\frac{P-\mu_{dd}}{\sigma}\right)^2}$$

where P is the father's phenotype. The exact same sum holds for the mother. For each child, the likelihood is similar but conditioned on the parent's genotypes marker, disease locus, and chance of recombination included (effectively a loop inside each possible parent genotype.) The reason we have to consider Dd and dD as separate in the parents is because this is where we are assuming D corredsponds to observed marker 1 and d with observed marker 2, but the other haplotype combination is possible as well, so in the summation these must also be included.

With the mother and father each having four possible haplotypes (given their marker genotypes) we are nesting the child possibilities across all 16 possible parental haplotypes. Within the 16 possible parental haplotypes, 3 possible cases for the disease haplotypes for the children arise.

In the first case the matings are not informative for linkage, but are still helpful in estimating all parameters aside from θ. These matings are $DD \times DD$, $dd \times dd$, and $DD \times dd$ (or $dd \times DD$). In this case the likelihood of each child's phenotype given their marker's phenotype is independent of which marker alleles they receive. Their likelihood nested under these parental mating types is

$$\frac{1}{4}\frac{1}{(2\pi\sigma^2)^{1/2}} e^{-\frac{1}{2}\left(\frac{P-\mu_f}{\sigma}\right)^2},$$

where μ_f represents the forced mean. In each of these matings the children's disease genotypes will be DD, dd, Dd, and dDd, respectively. That is, their disease genotypes are forced. Similarly for the other two cases for parental haplotypes in the case of doubly heterozygous (for the disease locus—$Dd \times Dd$, $Dd \times dD$, $dD \times dD$, and $dD \times Dd$) parents there are 16 possible child haplotype pairs. Each one will have a probability dependent on the familiar phenotypic likelihood given a disease genotype and the likelihood that each haplotype resulted from an intact disease-locus/marker-locus haplotypic transmission from parent to child or from a novel (recombined) haplotype. Then it is the case that each child has two haplotypes, one

derived from each parent. Each of the 16 possibilities then for each child has a different likelihood.

The last disease locus mating type has one heterozygous parent and one homozygous parent. There are 8 examples here: $DD \times Dd$, $DD \times dD$, $dd \times Dd$, $dd \times dD$, $Dd \times DD$, $dD \times DD$, $Dd \times dd$, and $dD \times dd$. This case is a hybrid of the first two cases with one of the disease locus haplotypes of the child being completely determined. Within each of the 8 matings above there are 8 possible disease locus/marker locus haplotype pairs. Each of these is accounted for in likelihood terms for each child. Finally, the family likelihood is calculated by multiplying every possible scenario of marker and disease locus haplotypes for every member of the family and summing the results over all families. This gives the likelihood for a given set of parameters μ_{DD}, μ_{Dd}, μ_{DD}, σ^2, q , and θ, given the data we observe. Because all of these parameters are unknown, they factor into the likelihood of each family to varying degrees.

The final step in maximizing this likelihood equation is to search $\mathbb{R}^6$ for the combination of parameter values that maximizes the complete likelihood data. One could do this over a grid of pre-specified increments, but this grid quickly expands as a power of 6 for each additional division of each parameter, making such search computationally infeasible for a high degree of discretization of the search space. This also only allows specific grid values to be chosen as optimal parameterizations and is therefore limited in its accuracy. Alternatively, one could use a method such as Newton-Raphson or expectation maximization to optimize this likelihood. The modern tools of computational intelligence provide more options when faced with such complex optimizations. In the next section, we describe the PSO algorithm used to successfully solve our linkage problem.

IV. Tools of Computational Intelligence

Recent years have seen a rise in the development and application of computational algorithms inspired by nature. For optimization problems, the most relevant of these are Evolutionary Algorithms (EA) [1], based in spirit upon observed processes which drive certain biological and ecological systems. The EA paradigm includes genetic algorithms [4] and swarm intelligence [2]. The core elements of EA are outlined as follows:

1. Initialize a population of agents in the search space
2. Apply evolutionary operations to the agents
3. Advance the population to the next generation

Depending on the problem domain, the agents can take a variety of representations. In the case of genetic algorithms, each agent is considered a "chromosome" and has the form of a solution to the optimization problem. For example, if we are optimizing a function of n variables, then the agent is a vector in $\mathbb{R}^n$. Governing the search are a number of evolutionary operations. Typical for genetic algorithms are operations categorized as *cross-over* and *mutation.*

In cross-over operations, two agents (called parents) are selected and their structure used to create an agent for the next generation (called the offspring.) In uniform crossover, the elements of the offspring correspond to those of the parents with equal probability. In one-point crossover, a selection point in the agent representation is chosen; all information to the left of the point is copied directly from one parent and all information to the right is copied directly from the other parent. Thus, the offspring retains entire sequences of data representation from the parents. One-point crossover can be extended to multi-point crossover, where any number of points are chosen and alternating strings from the parents are inserted wholesale into the offspring. In retaining information that proved the parent offspring were fit in their generation (where fitness is measured by some objective function), cross-over operations ensure useful information is passed on to the offspring so that they, too, may be fit. The end goal, of course, being to zero in on the combination of fitness genes with the highest utility for the given optimization problem.

Mutation operations, instead of copying information directly from the agents in the previous generation, modify the new agent in a way that may diverge significantly from the genetic information contained within the parents. In this way, the new generation is assured of an increase in chromosome diversity. This is the way EA's tackle the *exploitation vs. exploration* problem. We desire an algorithm which will exploit its current knowledge of the solution space in an effort to hone in on the optimal value. However, it is also advantageous for the algorithm to adequately explore said search space before settling on a final target. After all, what may seem like a global optimum in the first few generations may turn out to be nothing but the blip of a local optimum once a wider section of the search space is charted. Therefore, mutation operators are key to a successful EA implementation. Mutation can be as simple as the random shuffling of a given element of the offspring or as complicated as adding some measure of noise (Gaussian or otherwise) or some other nonlinear function to randomly selected parameters. There is a vast literature on the design and application of mutation operators [1]. For our purposes, this overview will suffice.

Genetic programming algorithms take the form of the general EA but instead of each agent representing the solution to an optimization problem, each agent is instead an encoding of a complete computer program, whose execution is evaluated by a fitness function. In this way entire programs can be evolved which will run and calculate the solution for a wide range of problems.

Other population-based algorithms are inspired by the emergent coordination seen among certain animals in nature. Ant colony optimization calculates the next generation of agents based on update equations which model the way ants lay a trail of pheromones to seek out stores of food and other desirable locations. Particle swarm optimization (PSO) [2] emulates the patterns of birds in flight, maintaining their

positions based on signals both social and individual in origin.

In PSO, the evolution operations take the form of two update equations—one for the influence of the other members of the swarm and one which reflects the experience of the individual agent itself. This combination of individual drive and social coordination is the basis for what turns out to be a quite powerful optimization algorithm.

The individual agents, or particles, of the PSO algorithm take the form of vectors in $\mathbb{R}^n$. In our case these are six-dimensional vectors. Their initial positions x_{ij} are distributed randomly throughout a given subset of $\mathbb{R}^n$. For our problem, the admissible ranges for the parameters are as follows: the means μ_{DD}, μ_{Dd}, and μ_{DD}, are locked between the largest and smallest value, σ^2 is bounded below by $.000001$ and above by the population standard deviation, the parameter q is in $[.000001, .5]$ as a value of 0 indicates we are no longer in a biallele system and any value greater than $.5$ is isomorphic to renaming our disease alleles from AB to BA, and the final parameter, θ, is allowed to range over $[0, .5]$, allowing our system to model a continuum of perfectly unlinked to perfectly linked genes.

In addition to a vector in $\mathbb{R}^n$ which indicates the particle's position in the solution space, each particle i also has a velocity v_i and personal best $pbest_i$. The velocity indicates the change in position this particle will undergo upon entering the next generation and the personal best stores the value of the particle's previous positions with the highest fitness level. A parameter maintained by the entire swarm called *global best* represents the maximum of the *pbests*. The social element of the PSO update, then, pulls the swarm towards *gbest* while the individual update zeroes in on *pbest*. The equations are given as follows:

$$ind_{ij}(t+1) = ind_{ij}(t) + c_1 r_1 (pbest_i(t) - x_{ij}(t))$$

$$soc_{ij}(t+1) = soc_{ij}(t) + c_2 r_2 (gbest_i(t) - x_{ij}(t))$$

$$v_{ij}(t+1) = ind_{ij}(t+1) + soc_{ij}(t+1)$$

where c_1 and c_2 are the individual and social constants and r_1 and r_2 are draws from a uniform random variable on $[0,1]$. Together, the use of current particle information and a random effect correspond to the crossover and mutation operators of the general EA's. Once velocity information is updated, the new positions are calculated by

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1)$$

In our implementation, we took parameter values of $c_1 = c_2 = .75$. We also constrain velocities to be within 85% of the total range of the given dimension. We use a ring topology where *gbest* is calculated based on the closest 6 particles, and set the swarm size to 30 particles.

An entire industry has sprung up to modify, flavor, and advance PSO algorithms in every imaginable direction. For further details and current research the interested reader is directed to [3] and [5]. Our results were achieved using a reasonably vanilla version of PSO in only a few minutes of computation time running in SAS on a desktop PC.

V. Conclusion

In this study we consider parametric linkage analysis in the case of a likelihood model with six parameters. To circumvent the problems historically associated with solving such models, we utilize the tools of computational intelligence to perform our optimization successfully.

Many biological systems are vastly more complex than the one governed by our six-parameter model. The fact that PSO scales well with dimension is important for these problems. Therefore, if the complexity of the likelihood function of a model can be effectively reduced through the introduction of more parameters (consistent with parametric linkage analysis) then these likelihoods, even though they will be of higher complexity, may be successfully optimized using tools such as those demonstrated herein.

The further development of biologically-inspired computational algorithms (in the EA vein) will also provide a boon to researchers studying actual biological systems. The symmetry here is impossible to miss. It is even the case that these algorithms themselves provide a framework in which to study the relevant biological system itself. Often it happens in engineering that a new application motivates a change to a given solution technique, only to discover that this change has significant utility beyond the original problem domain. The study of biological systems using nature-inspired algorithms is an area ripe for taking advantage of such positive externalities.

All that remains is for biological researchers to generate the complex models with many parameters whose likelihood can be coded using PSO or other evolutionary algorithmic approaches. One could even use a genetic algorithm to first come up with such a complex model with many parameters and then employ PSO to accurately estimate the parameters so that the likelihood of the observed data is then maximized. With data collection ongoing in the biological sciences we only need to be sure that we have many more data points than parameters. This is something that is becoming easier and easier to achieve in short generation time animals and plants, and this approach holds the potential to address complex and important biological issues.

References

[1] Meyer-Nieberg, S., & Beyer, H.G. *Self-Adaptation in Evolutionary Algorithms.* in *Studies in Computational Intelligence*. Springer-Berlin. 2007.

[2] Andries P. Engelbrecht, *Fundamentals of Computational Swarm Intelligence* . West Sussez, England: Wiley, 2003.

[3] Poli, R., Kennedy, J., & Blackwell, T. *Particle Swarm Optimization.* Swarm Intelligence. 1(1) pp 33-57. 2007.

[4] Shah, S., & Kusiak, A. *Cancer gene search with data-mining and genetic algorithms.* Computers in Biology and Medicine. 37(2) pp 251-261. 2007.

[5] Selvakumar, A.I., Thanushkodi, K. *A New Particle Swarm Optimization Solution to Nonconvex Economic Dispatch Problems.* IEEE Transactions on Power Systems. 22(1), pp 42-51. 2007.