

UC Riverside

UC Riverside Electronic Theses and Dissertations

Title

Applications of Genetic Algorithms in Bioinformatics

Permalink

<https://escholarship.org/uc/item/9087560g>

Author

Piserchia, Zachary

Publication Date

2018

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA
RIVERSIDE

Applications of Genetic Algorithms in Bioinformatics

A Thesis submitted in partial satisfaction
of the requirements for the degree of

Master of Science

in

Genetics, Genomics and Bioinformatics

by

Zachary T. Piserchia

December 2018

Thesis Committee:

Dr. Daniel Koenig, Chairperson

Dr. Connie Nugent

Dr. Jaimie Van Norman

Copyright by
Zachary T. Piserchia
2018

The Thesis of Zachary T. Piserchia is approved:

Committee Chairperson

University of California, Riverside

ABSTRACT OF THE THESIS

Applications of Genetic Algorithms in Bioinformatics

by

Zachary T. Piserchia

Master of Science, Graduate Program in Genetics, Genomics and Bioinformatics
University of California, Riverside, December 2018
Dr. Daniel Koenig, Chairperson

The field of Bioinformatics has advanced rapidly in recent years. Breakthroughs in sequencing technology have driven an explosion in the production of genetic data. As a result, the development of methods to interpret these data accurately and in a timely manner is a major bottleneck in bioinformatics research. This task is further complicated by imperfect knowledge of genetic features, as performance-altering assumptions can easily be introduced during algorithm development, and methodology may quickly become obsolete. Genetic algorithms are an evolution-inspired class of machine learning algorithms that show great promise to resolve these problems. These algorithms gradually refine solutions through natural selection, evolving a solution to a problem in bioinformatics rather than manually designing a search strategy. Due to this learning process determining how genomic features are identified, these genetic algorithms do not rely on human knowledge of the problem. Consequently, biases are largely limited to the data used to determine how fitness is evaluated. Genetic algorithms also make better use of computational resources by reducing search space and utilizing parallel computation. In order to examine the potential for genetic algorithms to facilitate research in the field,

this work explores many different implementations of these algorithms in bioinformatics. These genetic algorithm-based tools outperform existing methodology for many different problems, and their predictions can be used to guide experimental research.

TABLE OF CONTENTS

Abstract	iv
List of Figures	viii
Introduction	1
Genetic Algorithms Background	4
Genomics	11
Gene finding	11
Modelling Disease Risk	14
Multiple Alignment	18
Promoter Prediction	19
Transcription Factor Binding Sites	20
Determining Groups for Expression Analysis	23
Transcriptomics	25
Prediction of RNA Secondary Structure	25
Prediction of RNA-RNA Interactions	28
Discovery of Gene Regulatory Networks	29
RNA Editing Site Prediction	29
Proteomics	30
Protein Structural Prediction	31
Drug Discovery	32

Discussion	34
Advantages of Genetic Algorithms	34
Weaknesses of Genetic Algorithms	36
Conclusion	38
Bibliography	39

LIST OF FIGURES

Figure 1 - Crossover in Genetic Algorithms	9
Figure 2 - Mutation in Genetic Algorithms	10

Introduction

Growth in the field of DNA sequencing has generated a tremendous amount of genetic data, but much of it has only been superficially examined (Galperin and Koonin 2010). As a result, developing methods to decipher this information is one of the largest undertakings in the field of bioinformatics. Applications in bioinformatics largely focus on the discovery and annotation of genomic features from sequence data, as well as the characterization of genetic variants within populations (Manzoni et al. 2018) (Horner et al. 2010). While there is often a focus on genes, investigating regulatory elements such as promoter sequences is an equally important task. Problems in transcriptomics include the prediction of RNA structure from sequence data, the discovery of RNA interactions, and determining interactions between expressed genes (Manzoni et al. 2018).

Bioinformatics can be used in proteomics research to predict protein structures from sequence data but is also closely associated with pharmacology. Drug discovery and development can be greatly facilitated by computational predictions (Mandal, Johnson, Wu, and Bornemeier 2007). It is crucial for these areas of scientific research to accurately and efficiently analyze sequence data in order to provide meaningful predictions.

Despite the great value of these applications, interpreting sequence data is incredibly difficult. The extreme complexity and size of this data necessitates the use of computational approaches, but even these methods face challenges. Accuracy and runtime are principal concerns in the design of analysis algorithms, and improving one

often comes at the cost of the other (Ruffalo, LaFramboise, and Koyutürk 2011). In addition, traditional programming based on human knowledge of genetics problems would inherit the assumptions and biases of our currently imperfect understanding of genomic data.

Machine learning approaches present a strong opportunity to mitigate these problems. While this class of algorithms is not completely free of bias, they can be used to infer the rules required to identify genomic features from previous data. Consequently, the strength of machine learning to predict the presence and function of genomic structures is limited by data. Providing larger data sets reduces bias and error rate, improving the predictions of machine learning approaches (Brain and Webb 2000). It follows that as more genomic features become empirically verified, computational learning predictions will also improve. While many approaches to machine learning exist, three major approaches are Neural Networks (NNs), Support Vector Machines (SVMs), and Genetic Algorithms (GAs). Of these approaches, Genetic Algorithms are particularly suitable for bioinformatics applications, as they require little *a priori* knowledge of a problem they are applied to (Spears and De Jong 1990).

Neural networks are a machine learning method that is modeled after neurons in the brain. By using a series of interconnected nodes and weights representing action potentials, NNs can make decisions based on input data (Gómez-Ramos and Venegas-Martínez 2013). As this closely resembles the biological pathways for sensation and perception, NNs are extremely adept at image recognition and identifying patterns

(Schmidhuber 2015). Generally, NNs use supervised learning called backpropagation where a training set with a known solution is used to learn the weights of individual neurons. This method proceeds through several layers of neurons, and as such is a very slow process (Dharwal and Kaur 2016). However, once these weights have been learned, NNs can perform tasks extremely quickly. PilotNet, a 9-layer NN has been able to successfully steer and drive a vehicle without human input (Bojarski et al. 2017). In this example, the neural network uses pixels from video feed to determine the patterns constituting the road and traffic to steer the vehicle accordingly (Bojarski et al. 2017). Neural networks have been applied to problems in bioinformatics, but the runtime cost of training NNs is a significant constraint on throughput. NNs can also be combined with genetic algorithms to improve performance.

Support vector machines (SVMs) are a learning approach applied to classification problems. SVMs work through the use of training sets to calculate functions that separate data into two or more groups (Devi Arockia Vanitha, Devaraj, and Venkatesulu 2015). SVMs are very effective classifiers, and can be used to solve problems where data is not linearly separable through transformation (Winters-Hilt and Merat 2007). In bioinformatics, SVMs have been used to identify genes from unknown sequence data. GISMO is one such approach that uses a SVM in order to identify potential genes (Krause et al. 2007). GISMO learns oligonucleotide positions and frequencies (motifs) associated with genes in training data, and then uses this information to identify genes in unknown sequence data (Krause et al. 2007). This method can also be improved through the use of Genetic Algorithms to optimize initial tuning parameters.

Genetic Algorithms Background

Genetic algorithms (GAs) are an approach to machine learning with roots in evolution and genetics. Rather than relying on human knowledge to develop a program, GAs “evolve” a solution to problems using the principles of natural selection. Similar to evolution in biology, a population of algorithms is evaluated to determine individual “fitness.” Individuals in the population represent different parameters for a program, and their fitness is ranked based on how well the resulting program performed (Holland 1992). A simulation of reproduction is then used to produce the next generation, with high-fitness individuals contributing more to the gene pool (Beasley, Bull, and Martin 1993). Over many of these generations, the average fitness of the population gradually improves through evolution and the algorithm arrives at a solution (Beasley, Bull, and Martin 1993). While this general framework forms the basis for GA approaches, many different components of the algorithm must be considered in detail to successfully apply them to a problem.

The first step in GA development is to determine how the genotype of individuals in the population will be defined. Rather than using DNA, GAs use binary strings called “chromosomes” to represent an individual’s genes. These 0’s and 1’s are then translated into parameters and other components (i.e. genes) of a program, which are then used to solve the problem to which the GA is being applied (Holland 1992). For example, a chromosome in a pathfinding GA in robotics might include parameters such as the speed at which the robot will move, and at what distance from an obstacle it should be before

turning. In this case, the “genes” could be translated from binary to represent a decimal number for speed and/or distance. In order to do this, the binary values would need to be encoded and decoded (Haupt R. and Haupt S. 2004). Quantization is one method of encoding that takes non-overlapping values in a range and assigns them a unique binary value. This allows them to represent a larger range of values than pure binary, and also makes decimal numbers possible (Haupt R. and Haupt S. 2004). Choosing how to represent the problem and how many bits will be required for the chromosome is particularly important, as it will affect the performance of the algorithm where the maximum error will be equal to the intervals chosen during quantization (Haupt R. and Haupt S. 2004). Longer chromosomes may (or may not) be helpful in yielding a robust solution, but will greatly increase runtime costs. Conversely, shorter chromosomes may result in the GA running faster, but may result in a solution that is too simple and cannot be widely applied outside test data (Kim and Weck, 2005).

Populations in GAs consist of a static number of individual chromosomes, which are repeatedly replaced by those of the next generation. Population size must be carefully considered in this step, as small populations are extremely vulnerable to genetic drift (Rogers and Prugel-Bennett 1999). Because the forces of drift can undermine expected evolutionary changes and decrease fitness, a small population size or selecting too few parents can lead to drift and cripple learning in GAs (Rogers and Prugel-Bennett 1999). In contrast, using a large population size increases runtime costs. As GAs’ runtime is primarily derived from population size, finding a balance between learning rate and runtime is needed for a successful GA (Oliveto and Witt 2014).

While chromosomes define genotype in GAs, and mutation and crossover provide genetic variation, learning cannot occur unless we can evaluate phenotype in some way to derive a fitness value. This is accomplished through implementing a fitness function, which takes in an individual's chromosome and assigns it a fitness value (Manning, Sleator, and Walsh 2013). This involves translating the chromosome into parameters for a program (as described in chromosome design) and then actually running the program to arrive at a solution. The solution is then judged on its quality and accuracy to arrive at a fitness value. These determinations can be made in a variety of ways, but two of the most common are to rank members of the population in relation to the individual with the highest fitness, or to rank according to the average fitness of the population (Blickle and Thiele 1996). In a robotics example, a robot solving a maze might be assigned a fitness value based on how close it got to the exit, and the time it took for the robot to do so. This step is absolutely crucial for the success of a GA as it is the main learning step. It is also the most runtime intensive part of GAs as many individuals must be evaluated every generation.

Generations and repopulation are another core part of GAs, as they control the length of the learning period and how the gene pool of the next generation is formed. As a rule, the initial population of a GA starts with randomized chromosomes; this starts the population off with great diversity (Diaz-Gomez and Hougen 2007). To illustrate the importance of this rule, consider a graph where all possible solutions have their fitness ranked (a fitness landscape). This graph would have many hills and valleys, representing different solutions to the problem with varying degrees of effectiveness. If a GA were to

start with a monotypic population, there is potential for the algorithm to get stuck between two small local hills and miss a substantially larger global hill (Leung and Wang 2001). The algorithm would not explore solutions beyond the local peaks due to the fitness decline after leaving them. Consequently, the algorithm could accept a solution substantially less effective than the optimal (Leung and Wang 2001). Starting with a random population greatly ameliorates this problem by seeding many different “start points” for learning to branch from. This does not guarantee an optimal solution will be reached, but greatly increases the probability of reaching a near-optimal solution.

After fitness evaluations, reproduction occurs to create the population for the next generation. Although several variations of reproduction exist, the most common is to take the two individuals with the highest fitness to act as the parents for the next generation. Through crossover between the two parents, offspring chromosomes are generated until the population size is reached (Holland 1992). Single-point, two-point, and multi-point crossover can occur and similar to biological crossover, the parental chromosomes exchange material in regions adjacent to the crossover site(s) (see Figure 1) (Spears 1995). In order to maintain diversity and also allow for the introduction of novel genotypes, mutation can also occur. Mutation in GAs usually occurs at a fixed rate, and changes a single bit: a 0 becomes 1 or vice versa (see Figure 2) (Holland 1992). Choosing mutation rate is also an important part of GA implementation, as it affects the learning rate of the algorithm. This occurs as new solutions are explored through the introduction of novel genotypes produced by these mutations (Haupt, 2000). If the mutation rate is too low, it will take an extremely large number of generations to

thoroughly explore the solution space. Conversely, a mutation rate that is too high may result in fast initial learning rates, but fail in the long term by inhibiting convergence (Haupt, 2000). A final consideration is determining when to stop the algorithm and use the solution it has reached. Ensuring an accurate solution is found and mitigating runtime costs are both important factors in this decision, and this choice may vary based upon the problem. As such, several methods exist to terminate GAs, but the most common are setting limits to the number of generations and convergence (Bhandari, Murthy, and Pal 2012). Running for a fixed number of generations is a common approach, but also relies on the assumption that the algorithm has run for enough time to achieve a meaningful solution. The convergence method stops the algorithm once the population has become the same genotype and no new major changes occur, as this means all individuals have the same fitness and are no longer improving. Convergence is taken as evidence that the best solution has been found and the algorithm stops (Bhandari et al. 2012).

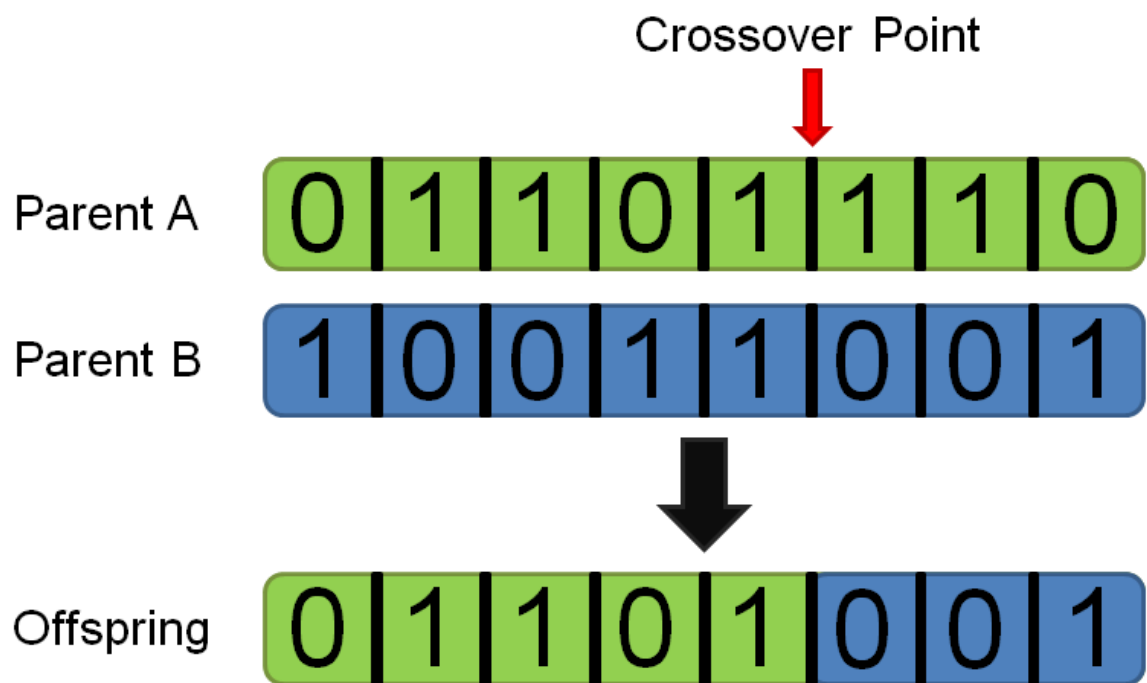


Figure 1: Single crossover occurs between two parents at a point selected at random to produce offspring

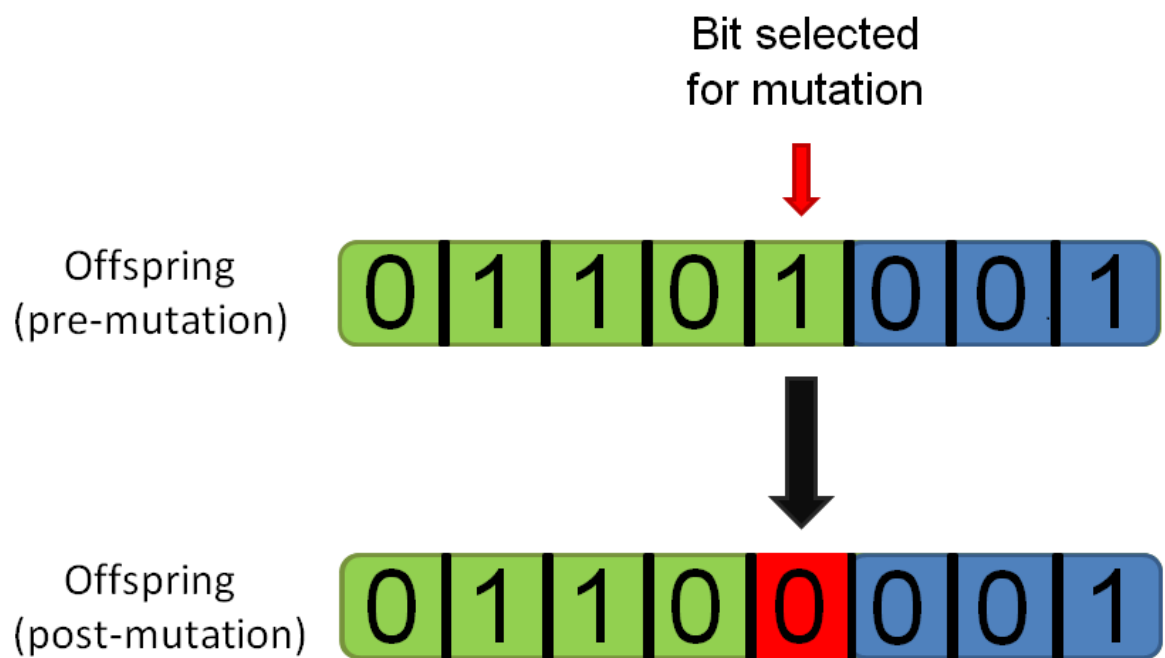


Figure 2: Mutation occurs after crossover,
resulting in a completely novel genotype

Genomics

The increased the pace of genomics research is largely due to massively parallel sequencing techniques, such as Illumina, which have resulted in a wealth of genetic data. Annotating and deciphering these new data is one of the greatest undertakings in the field, and genetic algorithms may greatly aid these efforts. GAs greatly benefit from the increased availability of data, drawing upon more potential knowledge to make their predictions. The automated nature of machine learning also requires minimal experimental research, greatly streamlining data analysis. This makes GAs an ideal approach for solving problems in genomics.

Gene-finding

One of the first steps in interpreting genome sequences is the identification of genes. Several models for gene prediction exist, including Position Weighted Matrices (PWMs), Spliced Alignment, and Hidden Markov Models (HMMs). PWMs align unknown sequences with those in a database to make gene predictions. By using conserved sequence sites at specific positions, the PWM can be used to predict gene features based on the alignment with functionally related sequences (Mathé, Sagot, Schiex, and Rouzé 2002). Glimmer3 (a gene-finding software used in prokaryotes) uses a PWM in order to identify ribosomal binding sites to help identify the start of a gene (Delcher, Bratke, Powers, and Salzberg 2007). Spliced alignment splits an unknown sequence into many pieces and aligns it to known sequences in a database. By

discovering similarities with known coding sequences, these alignments can be used to predict the presence of genes. However, this method suffers from inaccuracy in how the predictions are grouped. Often, a single gene may be split into several predictions, or multiple genes can be lumped together and predicted as one (Mathé et al. 2002). HMM approaches use training data to learn the probability of a particular base (A, T, C, or G) appearing at the current position in a sequence based on the identity of previous nucleotides (Mathé et al. 2002). As individual nucleotides often do not provide enough information, groups of nucleotides or “*k*-mers” are often used. If a *k*-mer at the current position is extremely improbable, it may represent a state transition (such as the sequence transitioning from exon to intron). Since exons and introns have statistically different nucleotide composition, these “hidden states” can be learned by the HMM. Through the use of training data sets, the algorithm can mark boundaries between coding and noncoding sequences to find genes (Mathé et al. 2002). Examples of HMM-based gene prediction software include GENSCAN (Burge and Carlin 1997), AUGUSTUS (Stanke and Morgenstern 2005), and HMMGene (Krogh 1997).

While all of these methods have successfully made gene predictions, most have several flaws. Current gene prediction software only finds transcribed regions of genes, and skips over or mislabels valuable regulatory features such as promoter sequences (Mathé et al. 2002). Comparative methods often struggle with the definition of “similarity;” as there is no general consensus on criteria for similarity, arbitrary guidelines are often used. This can result in comparative approaches predicting too many or too few genes, and identifying which genes are false positives is not readily apparent

(Mathé et al. 2002). Furthermore, it is difficult to rely upon evolutionary comparisons for the discovery of novel genes, as they may be substantially different from documented genes in a database. Long genes, long introns, and overlapping genes all pose difficulties in the developing gene prediction software, as they require identification of “special cases” (Mathé et al. 2002). Consequently, there is room for further improvement in gene-prediction through the use of GAs.

A GA approach to finding genes and annotating genomes was created by Chowdhury et al. (2016). Because over- or under- predicting genes often complicates the process of annotation, the problem was broken down into a sub-problem of finding exons (Chowdhury, Garai A., and Garai G. 2016). The authors decided to use homology with known exons in a database as a means to predict new exon boundaries from unknown genomic sequence data. To do this, chromosomes in the GA delineated possible exon boundaries, and fitness was evaluated using alignment scores to database exons (Chowdhury et al. 2016). The resulting approach was not only successful in a test case on human chromosome 21, but was also able to predict exons more accurately than GENSCAN, a well-known and widely used annotation tool (Burge and Carlin 1997). The success with this method demonstrates that the application of GAs may greatly advance efforts in the field.

A genetic algorithm approach by Hwang et al. (2013) can be used to predict which genes are essential. By definition, essential genes are required for an organism to live, meaning identifying them through lab experiments is difficult. While genome-wide

knockout experiments have been successful in identifying many essential genes, this process is extremely costly and time intensive (Hwang, Ha, Ju, and Kim 2013). Due to these issues, *in silico* prediction of essential genes is a desirable alternative. In order to accomplish this, the set of all an organism's genes must be classified into essential and non-essential groups. To do this, the authors chose to combine GAs with the area under a ROC (receiver operating characteristic) curve. ROC curves compare a classifier's true positive rate (Y axis) against false positive rate (X axis) for many different settings. As a high true positive rate and low false positive rate are required for accurate predictions, the area under the curve or AUC demonstrates how effective a given classifier is. For this reason, the authors chose to use AUC as a measurement of fitness in the GA. By using a training set of known essential and non-essential genes, the GA was successfully able to optimize a linear discriminant classifier (Hwang et al. 2013). The resulting GA_AUC classifier was highly successful, and outperformed other popular classification methods including logistic regression, polynomial kernels, RBF (radial basis function) support vector machines, and multilayer perceptrons (Hwang et al. 2013).

Modelling Disease Risk

While important, the discovery of genes and regulatory elements alone still presents incomplete genetic information. In order to learn more practical information in the form of phenotypic predictions, interactions between genomic features must also be considered. Due to the exponentially large number of possible interactions, as well as the multidimensional nature of the data involved, statistical models such as linear regression

have very little power to predict these associations (Moore, Hahn, Ritchie, Thornton, and White 2004). For the same reasons, these predictions are also extremely computationally intensive: solving this problem efficiently would require a reduction in the size of search space involved. GAs can be used to work around this problem as their populations explore many possible solutions to the problem, but are not a comprehensive search. For this reason, Moore et al. (2004) used GAs in order to develop a method for predicting models of disease risk.

In order to model interacting genotypes, penetrance functions can be used to predict the probability that a particular genotype will have a disease outcome. By manipulating these functions, a model first developed by Frankel and Schork (1996) can be used for multiple genes and alleles (Moore et al. 2004). Using this model, it is possible to predict disease risk for a given genotype if the set of penetrance probabilities is known. To calculate these values, as well as discover new and interesting models of disease risk, Moore et al. (2004) used a genetic algorithm with a fitness function that maximized the effect of gene interactions while minimizing the effect of individual genotypes. After running the resulting algorithm was successful at finding 1,000 different models for disease risk, including 3, 4, and 5 interacting gene models (Moore et al. 2004). The potential for GAs to make these complex predictions on phenotype shows great promise for the future of personalized medicine.

Another approach by Yang et al. (2004) combined GAs with a local search algorithm to study the influence of single nucleotide polymorphisms (SNPs) on disease

and predict genes associated with disease. This is a difficult problem, as multiple interacting SNPs may alter disease outcome, and SNPs with negligible effects on phenotype individually may collectively cause disease (Yang, Moi, Lin, and Chuang 2004). In addition, considering increased numbers of SNPs exponentially increases the computational complexity of the problem, to the point where traditional statistical methods become infeasible (Yang et al. 2004). Different types of genetic interactions must also be considered for accurate predictions; however, this increases complexity even further. For this reason, the authors investigated two main genetic models: a ZZ model, where two high risk alleles cause disease, and an XOR model, where heterozygosity at specific loci causes disease.

In order to discover interesting and novel patterns of disease, the GA's fitness function selected for SNPs that influenced disease more collectively, than they did individually. To further improve the method, the authors also included a local search algorithm to direct evolution within the GA. At the end of each generation, the local search was used to examine possible crossover exchanges; if an exchange would improve fitness, then it was carried out when creating the next generation (Yang et al. 2004). This combined approach had two advantages: the GA helped to mitigate the complexity of the problem and arrive at a solution faster, while the inclusion of the local search helped the GA escape local optima. Compared to a GA-only implementation, the local search GA was also found to improve the solution, finding more significant (based on chi-squared values) genetic models for disease (Yang et al. 2004). While this did improve the solution, the authors also noted that it did come with a slightly increased runtime cost, but

remained within the same order of approximation (Yang et al. 2004). It is likely that the discrepancy between the local search GA and regular GA increases with the complexity of model considered, making it a more suitable approach for complex interactions.

A final GA-based method for determining genes associated with disease was developed by Tahmasebipour et al. (2015). These researchers noted that the commonly used Genome Wide Association studies (GWAs) and comparative analysis approaches to disease gene prediction are limited by their focus on predicting single disease genes (Tahmasebipour K. and Houghten S. 2015). Because disease can be the result of many interacting alleles, SNPs that have a low individual contribution to disease risk may have a large effect in the presence of other specific alleles. In order to account for this possibility, the authors reframed the problem, representing disease association as a network of nodes representing genes, mRNA, proteins, and other agents (Tahmasebipour and Houghten 2015).

To do this, the authors implemented a GA that used a chromosome representing a set of disease genes. Fitness was evaluated using a criterion of “collaboration”: sets were ranked based upon interaction with other genes in the group, as well as interaction with known disease genes based on known protein-protein interaction (PPI) networks (Tahmasebipour and Houghten 2015). In order to ensure the algorithm did not discover the same networks, a limit was placed on how many known PPI genes could exist together in a single set. In this way, candidate disease genes are evaluated based on their association with known disease genes, using the “guilt by association” principle

(Tahmasebipour and Houghten 2015). To test the algorithm, the authors used breast cancer as an example, and the algorithm was able to successfully discover candidate disease genes. These predictions were validated by Genotator (Wall et al. 2010), a tool that ranks gene-disease associations based on clinical data. Several breast cancer gene candidates found by the GA overlapped with the top 1% of Genotator's candidates. In addition, the GA was able to discover many candidates not identified by Genotator (Tahmasebipour and Houghten 2015). The algorithm also had higher sensitivity than CIPHER (Guzman and D'Orso 2017), a leading method for the disease-gene association problem (Tahmasebipour and Houghten 2015).

Multiple Alignment

A separate, but equally important application of genomic data is to compare evolutionary relationships between taxa. To make these comparisons and draw meaningful conclusions, it is first necessary to align them. While programs like ClustalW (Thompson, Higgins, and Gibson 1994) and Multalin (Corpet 1988) can complete this task, there is still opportunity for the development and improvement of algorithms for multiple alignment. A method developed by Nizam et al. (2011) uses a cyclic GA to perform a multiple alignment (Nizam, Ravi, and Subbaraya 2011). By using a fitness function that scores aligned residues, the algorithm is able to solve for the gap positions. The resulting program was able to outperform ClustalW and Multalin by generating multiple alignments with better scores.

Multiple sequence alignment has a wide variety of applications, ranging from the construction of phylogenetic trees to comparing protein structure (Nizam et al. 2011). As these predictions are reliant upon the accuracy of the alignment, the improved Cyclic Genetic Algorithm Multiple Sequence Alignment (CGA-MSA) approach will facilitate efforts in these areas.

Promoter Prediction

The identification of genes is only a first step, as many other regulatory elements affect gene expression *in vivo*. Promoter sequences are particularly useful information in determining how a gene is expressed as they are directly associated with a downstream gene (or genes in operon structures). Prediction of these elements is a notoriously difficult problem due to high variability even within the same species. Furthermore, extrapolating information from one species to another is impeded by the presence of different regulatory architecture (Azad, Shahid, Noman, and Lee 2011). Search approaches using motifs such as the TATA box and initiators exist, but these approaches are prone to frequent false positives (Azad et al. 2011).

Previous approaches to promoter prediction have mainly focused upon the identification of select features or signals. The presence of TATA or CAAT boxes, CpG islands, known transcription factor binding sites, and pentamer motifs have all been used to predict the presence of a promoter in unknown sequence data (Xie, Wu, Lam, and Yan 2006). Various forms of pattern recognizing algorithms have been applied to the classification of promoters, including neural networks, discriminant analysis, and

individual component analysis. While they have had some success, these methods have struggled due to the fact that individual sequence patterns indicating promoters cannot be widely applied (Xie et al. 2006).

To combat these problems, Azad et al. (2011) developed a method combining genetic algorithms with a support vector machine (SVM) to classify sequences into promoter and non-promoter categories. In order to do this, a positive training set containing verified promoters, and a negative training set containing non-promoter sequences was provided to the SVM. By learning different weights for patterns (motifs) in the data, the SVM learns to predict whether an unknown sequence is a promoter or not. In order to determine which motifs should be used, the researchers used a GA. By using a fitness function that weighted the probability of a random triplet being found in the promoter dataset versus the non-promoter dataset, the GA determined which triplets served as the best indicator of a promoter sequence. As triplets alone have limited statistical power, the authors decided to use hexamers comprised of triplet pairs found by the GA (Azad et al. 2011). The resulting method outperformed existing methods, having a higher average sensitivity and specificity (Azad et al. 2011). The resulting PROMOBOT tool remains a cutting edge tool for plant promoter identification with an average sensitivity of 85% (Umarov and Solovyev 2017).

Transcription Factor Binding sites

In addition to promoter sequences, the identification of new Transcription Factor Binding Sites (TFBSs) is also an important source of regulatory information. The

discovery of these sites is particularly difficult as few have been experimentally verified, and the length of these sequences is particularly short (Jayaram, Usvyat, and Martin 2016). Due to this lack of data and the statistical limits that accompany short sequences, it is difficult to use comparative methods such as homology searches on their own. This possibility is further complicated by an unclear scope of sequence conservation: regions flanking binding sites may also suggest common ancestry (Levitsky et al. 2007). Most current TFBS prediction software use position weighted matrices (PWMs), which have been suggested to be a state-of-the-art method for this task (Jayaram et al. 2016). By scoring the probability of each base in particular positions of a sequence, PWMs can be used to identify “matches” to motifs that could indicate the presence of a binding site. While currently one of the best methods for TFBS prediction, PWMs presume sites to be independent even though neighboring nucleotides often matter. Furthermore, many of these methods are only able to identify clusters of TFBSs and not individual sites (Jayaram et al. 2016). Owing to these problems, improved methodology is necessary to move forward in TFBS prediction.

Due to the great difficulty in predicting TFBSs, Levitsky et al. (2007) developed a new GA-based method to predict binding sites. The authors noted that many PWMs assume that single or select sites are essential to TFBS functionality and their reliance on a few select bases results in a high false positive rate (Levitsky et al. 2007). In order to mitigate these problems, the authors decided to approach the problem in a different way, by identifying local dinucleotide pairings (LDPs) that could be potential sites before analysis. To do this, the GA’s population consisted of a set of LDP frequencies, and the

fitness function used a Markov model to select for local dinucleotide pairings that were statistically different from random sequence data (Levitsky et al. 2007). Once found, LPDs are tested using discriminant analysis to predict if an unknown sequence is a TFBS. A flexible window size also allows for flanking regions to be used; this can improve the strength of predictions through the inclusion of regulatory elements highly associated with TFBSs (Levitsky et al. 2007). This bottom-up approach to prediction was found to outperform widely used PWM methods, and had a lower false positive rate (Levitsky et al. 2007).

Since TFBSs are not the only class of short regulatory sequences, it is worth noting that these solutions have the potential to be applied more widely. Motif discovery is one of the largest classes of problems in bioinformatics, as the search for genomic features is largely guided by the presence of sequence motifs indicating their presence (Zara-Mirakabad et al. 2009).

Individual “monad” motifs are often the focus of research in this area, but more complicated dyad motifs (where one motif closely follows another) are often overlooked. This is likely a byproduct of the fact that many monad-focused approaches overlook dyads due to low individual contribution to the signal, despite their combined importance (Zara-Mirakabad et al. 2009). Dyads possess the advantage of more nucleotides, statistically strengthening predictions; this also provides the opportunity for spacers between the two motifs, as is often found in binding sites. For these reasons, Zara-Mirakabad et al. implemented and tested a genetic algorithm approach to finding dyad

motifs. Using chromosomes that outlined the positions of possible dyads, researchers evaluated fitness using a Position Frequency Matrix (similar to PWMs) to score the signaling strength of potential motifs (Zara-Mirakabad et al. 2009). The resulting algorithm was not only successfully able to predict dyads, but also outperformed existing dyad-prediction software such as AlignAce (Roth, Hughes, Estep, and Church 1998) and MITRA (Eskin and Pevzner 2002) (Zara-Mirakabad et al. 2009).

GA-based methods to TFBS prediction have outperformed the widely used position weighted matrices, resulting in an extremely useful research tool. This improved accuracy is particularly useful in identifying candidate TFBSs, which can be used to direct experimental research. In addition, GA-based predictions do not require the use of comparative methods, allowing them to be applied where data based on evolutionary conservation is unavailable. This is particularly helpful in discovering TFBSs, as there is currently little experimentally verified binding data to make comparisons across multiple species. The advantages of GAs circumvent many previous obstacles in discovering TFBSs, providing new opportunities for regulatory research.

Determining Groups for Expression Analysis

In order to translate sequence data into meaningful phenotypic predictions, it is important that specific motifs be linked to *in vivo* mRNA expression. The advent and increasing accessibility of techniques such as RNA-seq as well as other methods such as microarrays have made this possible, but deciphering these data can be difficult (Ooi, and Tan 2003). Different genes may contribute positively or negatively towards a

particular phenotype, and some may have a stronger influence than others. It is also possible for a single gene to influence several different phenotypes. Analysis of mRNA expression data therefore requires these different “classes” of genes to be organized by sign and strength with respect to a particular phenotype.

This is a particularly difficult problem, as the number of possible combinations of groupings grows exponentially as more genes are analyzed. Traditional mathematical methods have struggled to address this problem; there is substantial drop-off in accuracy for these methods when more than two or three groups are predicted for expression data (Ooi et al. 2003). Since differences in expression can affect many different biological pathways, the two or three classes provided by these methods are insufficient for meaningful predictions to be made. To accurately represent biological data, methods for expression analysis must be able to organize expressed genes into many different classes. In order to do this, Ooi et al. (2003) developed a method combining GAs and maximum likelihood to predict classes of genes from expression data, in order to determine tumor phenotype. The authors used a chromosome representing different sets of genes, and evaluated fitness with a maximum likelihood classifier. The classifier in turn used a training set containing multiple tumor samples, allowing the gene classes to be scored based on the probability that they would result in tumor growth (Ooi et al. 2003). Implementing the program with a GA also resulted in several unique advantages: the optimal group size for classifying the expression data was solved alongside the membership of the group; and the use of a population allowed for a parallel search for multiple possible gene classes at the same time (Ooi et al. 2003). The resulting

GA/MLHD program was more accurate than existing expression class prediction methods, resulting in an improved means of analyzing expression data (Ooi et al. 2003). This algorithm has been applied to microarray data from multiple cancer cell lines, suggesting GAs can provide practical diagnostic applications in the field of medicine (Ghaheri, Shoar, Naderan, and Hoseini 2015).

Transcriptomics

Once protein structures have been predicted and even verified, it is still crucially important to consider the effects of regulation to make accurate *in vivo* predictions. Due to the effects of regulatory non-coding RNAs such as miRNA, coding sequences alone provide an incomplete view of gene expression (Reuter and Mathews 2010). Despite their importance, approaches to determine RNA structures require tremendous effort. In the case of ribosomal RNA secondary structure, predictions were slowly improved over 20 years of research before being validated by crystal structure (Dowell and Eddy 2006). The use of machine learning approaches to predict transcripts and RNA structures can greatly facilitate this process: isolation and verification are much easier once the basic structure is known.

Prediction of RNA Secondary Structure

Previous attempts at RNA secondary structure predictions have mainly used comparative analysis and thermodynamics based methods (Gardner and Giegerich 2004). These approaches are similar to those used in protein structural prediction; comparative

analysis uses sequence homology to predict structure, while thermodynamics methods seek to find the lowest free energy conformation of RNA secondary structure (Gardner and Giegerich 2004). Many thermodynamics methods also struggle due to their reliance on the assumption that the RNA alignment is correct; if this is not the case, their predictions will be inaccurate (Dowell and Eddy 2006). This is likely the reason why the accuracy of thermodynamics methods is lacking, with 73% accuracy at best for even short sequences of RNA (Reuter and Mathews 2010). This number drops even further when applied to longer sequences. Comparative analyses have also been applied, and are known as one of the most accurate current methods (Dowell and Eddy 2006). These analyses use a multiple alignment between homologous RNAs, using sequence conservation to predict structural similarity. Despite this, they require an extremely large number of sequences in order to find sufficient conservation to accurately identify folding sites (Reuter and Mathews 2010). Sankoff algorithm approaches have also been applied to the problem of RNA structural alignment (Havgaard and Gorodkin 2014). These methods simultaneously compute RNA alignment and folding to predict a consensus structure. However, this approach has mostly been limited to pairwise alignments due to prohibitive computational costs (Dowell and Eddy 2006).

While alignments may aid in the prediction unknown RNA secondary structures, this method is insufficient on its own. It does not consider that even small changes in RNA sequences can affect structure and stability. It follows that it is possible for even high-scoring alignments to have very different structures *in vivo*. At the same time, alignments provide a means to predict RNA secondary structure without experimental

validation (Notredame, O'Brien, and Higgins 1997). As *ab initio* methods are currently lacking due limited understanding of RNA folding *in vivo*, there is opportunity to improve the method.

A hybrid method combining alignment and thermodynamics might improve RNA structure predictions. Notredame et al. (1997) implemented this with PRAGA, a genetic algorithm-based approach that combines structural alignments with dynamic programming and free energy to predict RNA secondary structure. The algorithm begins with some dynamic programming in order to create a main “backbone” for the alignment. This creates an accurate main alignment to which finer tuning can be applied by the GA. The use of the GA reduces the search space of the problem as not all alignments are scored, minimizing expensive fine-scale calculations (Notredame et al. 1997). Combined with the dynamic programming backbone, this allows the algorithm to still make accurate predictions with relatively low runtime. In addition, the reduction in complexity allows the algorithm to run on longer alignments than other methods (Notredame et al. 1997). To compensate for the reliance on previously verified RNA structures in the alignment portion of the algorithm, researchers also included thermodynamics to score the chemical stability of the resulting structure. Again, in order to reduce the runtime cost of these calculations, this was combined with alignment score in the fitness function for the GA. Due to these improvements, PRAGA was not only able to successfully predict RNA secondary structures *in silico*, but also succeeded in discovering pseudoknots, which many previous methods have failed to find (Notredame et al. 1997). PRAGA established GAs as a means of predicting RNA structure, and served as the backbone for the newer

Consensus Folding GA (Cofolga2) algorithm (Taneda 2008). Cofolga is able to run faster than contemporary methods for RNA secondary structure prediction while maintaining high sensitivity (Taneda 2008).

Prediction of RNA-RNA interactions

In order to accurately predict expression, it is necessary not only to understand RNA structures but also to determine which structures interact. Many forms of RNA are regulatory in nature, and modify expression levels through hybridization with other transcripts. For example, miRNAs bind to specific mRNA transcripts and mark them for degradation (Bhaskaran and Mohan 2014). These interactions are also of interest due to the potential for novel disease treatments via gene therapy. Previous algorithms that have attempted to solve this problem have struggled with incredibly high runtime requirements due to the exponential complexity of RNA-RNA conformations (Montaseri, Zare-Mirakabad, and Moghadam-Charkari 2014). In order to improve the efficiency of this process, Montaseri et al. (2014) developed a GA to predict interactions between two RNA molecules. By determining fitness using a free energy function, the author's GA was able to predict RNA-RNA interactions with accuracy comparable to currently existing methods (Montaseri et al. 2014). In addition, the program was able to arrive at these solutions with less runtime. Due to the runtime bottleneck in this field, this means the application of GAs can expedite the slow process of identifying RNA-RNA interactions.

Discovery of Gene Regulatory Networks

While expression data is extremely useful in predicting phenotype, it can also be used to discover patterns of gene regulation. For this reason, Sîrbu et al. (2010) reviewed several approaches to finding Gene Regulatory Networks (GRNs) from DNA microarray data. They implemented and tested seven different evolutionary algorithms (a broader class of algorithms related to GAs) for this task. Of these approaches, artificial neural networks (ANN) combined with GAs and a genetic algorithm for local search (GLSDC) performed the best (Sîrbu, Ruskin, and Crane 2010). The ANN-GA made the most accurate predictions for small networks, and also was able to arrive at a solution with the least amount of runtime. In a test case for noisy expression data, ANN-GA and GLSDC continued to outperform other methods and was still able to provide accurate results (Sîrbu et al. 2010). These methods not only provide a framework for the discovery of gene regulatory networks, but also showcase the ability for hybrid GA methods to outperform their constituent parts.

RNA Editing Site Prediction

A final regulatory process to consider is the possibility of post-transcriptional modification of RNA. Of these possible alterations, RNA editing is currently not very well characterized; in fact, many known instances of this process were discovered by accident (Thompson J. and Gopal S. 2006). Consequently, there is opportunity for computational methods to aid in the search for possible editing sites and direct

experimental validation efforts. In order to discover new RNA editing sites, Thompson et al. (2006) implemented a GA that predicts C → U editing sites in plant mitochondria. In order to make these predictions, candidate sites had fitness evaluated based on comparison with known cytosine editing sites (Thompson and Gopal 2006). The resulting RNA Editing site prediction by Genetic Algorithm Learning (REGAL) algorithm was able to successfully predict C → U editing sites with 87% accuracy in testing, outperforming existing methods (Thompson and Gopal 2006). While this is not perfect, this tool could be applied to discover new RNA editing sites, which in turn could further improve the algorithm's predictions. Collaboration between computational and experimental research in this field could prove vital in learning more about this biological process.

Proteomics

Another major goal of genetics and bioinformatics research is to determine the resulting protein from coding sequences in the genome. While it is simple to translate coding DNA into amino acid sequences, the process of predicting the resulting protein structure is an incredibly complex problem. This stems from the fact that the number of possible rotation angles in an AA chain results in a combinatorial explosion of conformations. Of all these possible options, only a few will be chemically stable, and of these even less may be present *in vivo*.

Protein Structural Prediction

To eliminate complications that stem from variability in chemical stability, a free energy function, such as Shannon entropy, is often used to evaluate how viable a protein structure would be. However, to calculate this for all possible structures would be infeasible due to the immense computational cost. In order to work around this complexity, some methods have used simplifications of protein surfaces, side chains, and even main chains to accelerate the process, but this comes at a great cost to accuracy (Pedersen J. T. and Moult J. 1997). A separate strategy called correlated mutation analysis (CMA) limits the number of comparisons by focusing on residues at conserved positions (Bywater 2016). While this is a leading approach in protein structural predictions, it has several drawbacks. CMA assumes coevolution between the compared sequences based only upon homology which can be incorrect, as similar structures may have evolved independently. An additional problem is that sequence similarity does not always indicate structural similarity (Bywater 2016).

In order to maintain high accuracy and minimize assumptions, Pedersen and Moult (1997) used genetic algorithms to reduce the search space of the problem. By doing so, fewer protein structural evaluations would be required, mitigating computational costs while maintaining accuracy. The authors used a fitness function comprised of a free energy model, such that the GA's population would automatically evolve towards structures that were chemically viable (Pedersen and Moult 1997). This eliminates a massive number of calculations on unstable protein structures that could not

exist long enough *in vivo* to have a meaningful biological effect. While the resulting algorithm was unable to run faster than a state of the art Monte Carlo Chaining (MCC) approach, it outperformed MCC in structural prediction as resulting structures had a lower free energy (Pedersen and Moult 1997). Furthermore, the authors mention that parallelization of the algorithm is a possibility, suggesting there is a potential for GAs to outperform existing methods in runtime as well (Pedersen and Moult 1997).

A later approach developed by Rashid et al. (2013) sought to further improve existing methods by mixing low- and high-resolution energy models in a GA. By including the lower resolution Hydrophobic-Polar model, the algorithm was able to explore conformations with hydrophobic cores. This is a problem previous approaches have struggled with, as hydrophobic cores are often rejected early on due to lower initial free energy before the rest of the structure is found and considered (Rashid, Newton, Hoque, and Sattar 2013). Later on, a high resolution model is used to improve the accuracy of larger structures. This allowed the approach to explore more possible conformations without sacrificing accuracy. Although computationally intensive, this approach was able to outperform existing methods by making lower free energy structural predictions with lower root mean square deviation values (Rashid et al. 2013). These GA-based *in-silico* structural predictions provide an opportunity to explore possibilities for pharmaceutical compounds without costly laboratory work.

Finally, a GA-based approach to predicting protein structure implemented by Huang et al. (2004) further improved upon previous approaches. The authors pointed out

that hydrophobic-hydrophilic models (HP model) of protein folding had implicit assumptions caused by thermodynamics. Due to the fact that free energy is lower for hydrophobic cores surrounded by hydrophilic amino acids, there is a tendency for these HP models to gravitate toward these structures without first exploring other possibilities (Huang, Yang Chang-Biau, Tseng, and Yang Chia-Ning 2004). To mitigate this bias, the authors decided to use a 3D lattice model representation to predict protein structure, using GA chromosomes to represent different possible conformations of the protein. To further broaden the scope of possible structural predictions, the authors included many different chemical properties based on sequence homology to inform their predictions (Huang et al. 2004). In order to do this, a fitness function was implemented to consider many different properties that evaluate candidate protein conformations. These properties include secondary structures, such as alpha helixes and beta sheets, side chains, disulfide bridges, and electrostatic interactions, and were all used to score the chemical viability of possible structures (Huang et al. 2004). By using homology to discover and include these properties, the resulting algorithm was able to make more accurate predictions than those of other HP model based methods (Huang et al. 2004).

Drug Discovery

Protein structural predictions are important for discovering the biological effects of genes, but are also critically important for drug development. Searching for new compounds with desirable bioactivity is of great medical and economic importance. However, screening for new bioactivity is an extraordinarily costly and time consuming

process, meaning experimental investigation of every compound is infeasible (Katsila, Spyroulias, Patrinos, and Matsoukas 2016). As such, computer-based methods to determine which compounds to screen experimentally are extremely important for the field. Still, the exponentially large number of possible combinations and problem of local optimums requires a systemic search in order to discover the best drug candidates (Mandal et al. 2007). A recent approach developed by Mandal et al. (2007) uses GAs in order to solve this problem. The authors chose to represent the problem with a chromosome comprised of chains and substructures of a possible compound and evaluated fitness using a simulation of activity for the desired target (Mandal et al. 2007). Due to the generational improvement in GAs, this also greatly reduces the search space by improving off existing chains and combinations in the population rather than creating all compounds from scratch. The integration of computer-based predictions can greatly facilitate drug design cut research costs (Katsila, et al. 2016). By reducing search space, GAs provide a novel, efficient way to screen drug candidates.

Discussion

Advantages of Genetic Algorithms

Genetic algorithms are a powerful, flexible tool to address many challenges in bioinformatics. Any problem where solutions can be ranked in a meaningful way to measure fitness can be solved through the evolutionary aspect of GAs. This is particularly helpful as many other machine learning methods are reliant on the use of

training data. These methods often fall victim to biases in the data, making selection of the training sets a problem itself (Kubat and Matwin 1997). GAs only require a fitness function and therefore avoid these problems; these functions can often be designed with limited prior knowledge of a given bioinformatics problem. This makes GAs ideal for problems where the search space is poorly understood, as they can tolerate some noise in the fitness function (Manning et al. 2013). Due to these traits, GAs can be rapidly applied to new areas of bioinformatics research where knowledge is imperfect.

This flexibility also leads GAs to be easily hybridized with other approaches. In many cases, combining pre-existing methods with GAs can result in even better algorithms, as demonstrated by the work of Hwang et al. (2013), Notredame et al. (1997), Sîrbu et al. (2010), and Yang et al (2004). In addition to overall improvements through combined methods, hybrid approaches may also be used to focus on particular aspects of a solution that may be desirable. In this sense, specialized methods that may be weak on their own can be improved with the strength of GAs while maintaining their mastery in particular niches.

Another particularly useful aspect of GAs is their ability to reduce the computational complexity of a given problem. Due to the fact that they explore only some, but not all solutions for a given problem, the number of calculations, and therefore runtime, required is greatly diminished. Complexity can be reduced even further through the use of variable reduction strategies; these incorporate some domain knowledge to greatly improve the performance of a GA (Wu et al. 2013). This is extremely helpful in

bioinformatics, as most problems in the field involve complex statistics to arrive at a solution. This advantage can be even further improved through parallelization; the population-based system of GAs can easily be distributed over multiple processors or cores greatly reducing runtime requirements (Manning et al. 2013).

Perhaps the greatest and most straightforward advantage of genetic algorithms is their ability to learn solutions with little knowledge of the problem beforehand. Even simple definitions of fitness may result in extremely complex and effective solutions to a problem (Angeline, P. J. 1994). Owing to GAs' roots in evolution, emergent solutions can arise, and just as in nature the sum of individually evolved parts may result in a greater overall complexity than is apparent for each piece. As a result, the selective process in GAs often results in robust solutions that would not have been foreseen by human-directed research. These approaches also limit assumptions by researchers, as the learning process can take place without providing the algorithm with explicit knowledge (Angeline, P. J. 1994).

Weaknesses of Genetic Algorithms

While genetic algorithms are a very effective tool in bioinformatics, no method is perfect and GAs have some drawbacks. While GAs often use less *a priori* knowledge than other approaches, they do require researchers to design a fitness function (Spears and De Jong 1990). Some problems may require complex fitness functions, meaning substantial knowledge of the problem may still be required. The fitness function is also instrumental in guiding the algorithm to a solution; biases can easily be introduced to the

function that alter the program's output. If fitness is poorly defined or approximations are introduced, the selection process will still follow those rules resulting in error (Sastry and Goldberg 2002). This will result in a poor solution that may only work for a test case, and fail in "real world" applications. Decisions on population size, number of generations, and crossover and mutation rates may also influence learning and require tuning for the best results (Manning et al. 2013). Unlike the fitness function, these decisions must all be made arbitrarily, resulting in a painstakingly trial-and-error process for tuning parameters. Optimizing GAs can be a problem in itself, and it can require the algorithm to be run several times to find the best solution (Manning et al. 2013).

Another concern is that while GAs can be used to find strong solutions, they are generally only "answers." In other words, their output does not directly teach scientists the rules for the problem. As discovering underlying genetic methods and discovering how genomes are organized is a principal topic of research, this is one of the greatest shortcomings of GAs and machine learning in the field. GAs can be used to guide this type of research by providing examples, but only provide answers and never an explanation as to why the algorithm arrived at it. In this sense, GAs are a "black box" with an input and output (Manning et al. 2013).

In the scope of bioinformatics methodology, GAs are relatively fast for making accurate predictions. However, the fact that a large population size and many generations are required by GAs means they still take a substantial time to run; they are generally slower than directed search methods (Manning et al. 2013). These runtime constraints

mean GAs are not the best option for simple problems that do not require substantial machine learning to solve. In cases where the rules of the problem are well established, a search algorithm incorporating pre-existing knowledge may be more efficient (Manning et al. 2013). An “educated guess” solution may also suffice in some cases, meaning the improved accuracy of a GA may not be worth the runtime cost. In these situations GAs would take an extremely long time to run compared to other algorithms, making traditional methods more suitable. The runtime costs of GAs can be mitigated by parallelization, but they can still be considerably slow (Manning et al. 2013).

Conclusion

Genetic algorithms provide an incredibly diverse and effective set of tools for bioinformatics analysis. Their ability to solve major problems in genomics, transcriptomics, and proteomics may greatly expedite bioinformatics research. This is demonstrated by the consistently higher performance of GA methods over existing methods in these categories. GA methods are also highly adaptable, and can often overcome weaknesses through the use of hybrid approaches. Finally, additional sequence data will continue to improve GAs as more learning examples become available. The accuracy, efficiency, and potential for growth in GA-based methodology provides a robust solution to data analysis in bioinformatics.

Bibliography

- Angeline, P. J. 1994. Genetic Programming and Emergent Intelligence. Advances in Genetic Programming. Ed. K. E. Kinnear. Cambridge, MA. MIT Press.
- Azad A. K. M., Shahid S., Noman N., and Lee H. 2011. Prediction of plant promoters based on hexamers and random triplet pair analysis. *Algorithms for Molecular Biology* 6:19.
- Beasley D., Bull, D. R., and Martin R. R. 1993. An Overview of Genetic Algorithms: Part 1, Fundamentals. *University Computing*, 15(2) pp. 58-69.
- Bhaskaran, M., & Mohan, M. 2014. MicroRNAs: History, Biogenesis, and Their Evolving Role in Animal Development and Disease. *Veterinary Pathology*, 51(4), 759–774. <http://doi.org/10.1177/0300985813502820>
- Bhandari D., Murthy C. A., and Pal S. K. 2012. Variance as a stopping criterion for genetic algorithms with elitist model. *Fundamata Informaticae* 120:145-164. doi: 10.3233/FI-2012-754
- Blickle T. and Thiele L. 1996. A Comparison of Selection Schemes Used in Evolutionary Algorithms. *Evolutionary Computation*, 4-4:361-394. Dec. 1996. doi: 10.1162/evco.1996.4.4.361

- Bojarski M., Yeres P., Choromanska A., Choromanski K., Firner B., Jackel L., and Muller U. 2017. Explaining how a Deep Neural Network trained with End-to-End Learning steers a car. arXiv:1704.07911 [cs.CV]
- Brain, Damien & Webb, Geoffrey. 2000. On the Effect of Data Set Size on Bias and Variance in Classification Learning. *Proceedings of the Fourth Australian Knowledge Acquisition Workshop*.
- Burge, C. and Karlin, S. 1997. Prediction of complete gene structures in human genomic DNA. *Journal of Molecular Biology* 268:78-94.
- Bywater, Robert P. 2016. Comparison of algorithms for prediction of protein structural features from evolutionary data. *PLoS ONE* 11(3):e0150769.
doi:10.1371/journal.pone.0150769
- Chowdhury B., Garai A., and Garai G. 2016. An optimized approach for annotation of large eukaryotic genomic sequences using genetic algorithm. *bioRxiv* preprint, posted online Oct. 25, 2016.
- Corpet F. 1988. Multiple sequence alignment with hierarchical clustering. *Nucl. Acids Res.*, 16 (22), 10881-10890.
- Delcher A., Bratke K., Powers E., and Salzberg S. 2007. Identifying bacterial genes and endosymbiont DNA with Glimmer. *Bioinformatics* 23:673–679.
doi:10.1093/bioinformatics/btm009

- Devi Arockia Vanitha C., Devaraj D., Venkatesulu M. 2015. Gene Expression Data Classification using Support Vector Machine and Mutual Information-based Gene Selection. *Procedia Computer Science* 47:13–21. doi: 10.1016/j.procs.2015.03.178
- Dharwal R. and Kaur L. 2016. Applications of Artificial Neural Networks: A Review. *Indian Journal of Science and Technology*, 9(47). doi:10.17485/ijst/2016/v9i47/106807
- Diaz-Gomez P. A. and Hougen D. F. 2007. Initial population for genetic algorithms: A metric approach. *International Conference on Genetic and Evolutionary Methods*, 2007. Las Vegas, Nevada, USA.
- Dowell R. and Eddy S. 2006. Efficient pairwise RNA structure prediction and alignment using sequence alignment constraints. *BMC Bioinformatics* 7:400. doi:10.1186/1471-2105-7-400.
- Eskin E., and Pevzner P. 2002. Finding composite regulatory patterns in DNA sequences. *Bioinformatics* 18:354-363.
- Frankel WN and Schork NJ. 1996. Who's afraid of epistasis? *Nature Genetics* 14:371-373. [Pubmed 8944011]

- Galperin M. and Koonin E. 2010. From complete genome sequence to “complete” understanding? *Trends in Biotechnology* 28(8):398–406.
doi:10.1016/j.tibtech.2010.05.006.
- Gardner P. P. and Giegerich R. 2004. A comprehensive comparison of comparative RNA structure prediction approaches. *BMC Bioinformatics* 5:140.
<https://doi.org/10.1186/1471-2105-5-140>
- Ghaheeri A., Shoar S., Naderan M., and Hoseini S. S. 2015. The Applications of Genetic Algorithms in Medicine. *Oman Medical Journal* 30(6), pp. 406–416.
<http://doi.org/10.5001/omj.2015.82>
- Gómez-Ramos E. and Venegas-Martínez F. 2013. A Review of Artificial Neural Networks: How well do they perform in Forecasting Time Series? *Journal of Statistical Analysis* 6(2): 7-15.
- Guzman C. and D'Orso I. 2017. CIPHER: A flexible and extensive workflow platform for integrative next-generation sequencing data analysis and genomic regulatory element prediction. *BMC Bioinformatics* 18. doi:10.1186/s12859-017-1770-1.
- Haupt, R. L. 2000. Optimum Population Size and Mutation Rate for a Simple Real Genetic Algorithm that Optimizes Array Factors. *Applied Computational Electromagnetics Society Journal* 15: 1034-1037. 10.1109/APS.2000.875398.

- Haupt R. L. and Haupt S. E. 2004. Practical Genetic Algorithms. Second Edition. Hoboken, NJ: John Wiley & Sons, Inc. Chapter 2 pp. 27-50.
- Holland, John. 1992. Genetic Algorithms. Scientific American July 1992:66-72
- Horner D. S., Pavesi G., Castrignanò T., De Meo P. D., Liuni S., Sammeth M., Picardi E., and Pesole G. 2010. Bioinformatics approaches for genomics and post genomics applications of next-generation sequencing. *Briefings in Bioinformatics* 11:2, pp. 181–197. <https://doi.org/10.1093/bib/bbp046>
- Hwang K., Ha B., Ju S., and Kim S. 2013. Partial AUC maximization for essential gene prediction using genetic algorithms. *BMB Reports* 46:41-46. doi: 10.5483/BMBRep.2013.46.1.159
- Huang Y., Yang Chang-Biau, Tseng K., and Yang Chia-Ning. 2004. Protein Folding Prediction with Genetic Algorithms (Master's Thesis). Retrieved from *Research Gate*: https://www.researchgate.net/profile/Chia-Ning_Yang/publication/240836700_Protein_Folding_Prediction_with_Genetic_Algorithms/links/004635296a5ec9a5b0000000/Protein-Folding-Prediction-with-Genetic-Algorithms.pdf
- Jayaram N., Usvyat D., and Martin A. C.R. 2016. Evaluating tools for transcription factor binding site prediction. *BMC Bioinformatics BMC Series* 1298. doi:10.1186/s12859-016-1298-9.

- Jürgen Schmidhuber. 2015. Deep learning in neural networks: An overview. *Neural Networks* 61:85-117. <http://dx.doi.org/10.1016/j.neunet.2014.09.003>
- Katsila T., Spyroulias G. A., Patrinos G. P., and Matsoukas M. 2016. Computational approaches in target identification and drug discovery. *Computational and Structural Biotechnology Journal* 14, pp. 177-184.
<https://doi.org/10.1016/j.csbj.2016.04.004>
- Kim I. Y., and Weck O. L. 2005. Variable chromosome length genetic algorithm for progressive refinement in topology optimization. *Structural Multidisciplinary Optimization* 29:445-456. doi: 10.1007/s00158-004-0498-5
- Krause, L., McHardy, A. C., Nattkemper, T. W., Pühler, A., Stoye, J., and Meyer, F. 2007. GISMO—gene identification using a support vector machine for ORF classification. *Nucleic Acids Research* 35(2), pp. 540–549.
<http://doi.org/10.1093/nar/gkl1083>
- Krogh, A. 1997. Two methods for improving performance of an HMM and their application for gene finding. *Proceedings of the 5th International Conference of Intelligent Systems for Molecular Biology*. pp. 179-186. AAAI Press, Menlo Park, CA.
- Kubat M. and Matwin S. 1997. Addressing the Curse of Imbalanced Training Sets: One-Sided Selection. *Proceedings of the Fourteenth International Conference on Machine Learning*. pp. 179-186, San Francisco, CA, July 8-12 1997.

- Leung Y. and Wang Y. 2001. An orthogonal genetic algorithm with quantization for global numerical optimization. *IEEE Transactions on Evolutionary Computation*, 5-1:41-53. Feb 2001. doi: 10.1109/4235.910464
- Levitsky V. G., Ignatieva E. V., Ananko E. A., Turnaev I. I., Merkulova T. I., Kolchanov N. A., and Hodgman T.C. 2007. Effective transcription factor binding site prediction using a combination of optimization, a genetic algorithm and discriminant analysis to capture distant interactions. *BMC Bioinformatics* 8:481. doi:10.1186/1471-2105-8-481.
- Mandal A., Johnson K., Wu, C. F. J., and Bornemeier D. 2007. Identifying promising compounds in drug discovery: genetic algorithms and some new statistical techniques. *Journal of Chemical Information and Modelling* 47:981-988.
- Manning T., Sleator R. D., and Walsh P. 2013. Naturally selecting solutions: The use of genetic algorithms in bioinformatics. *Bioengineered* 4:266-278.
<http://dx.doi.org/10.4161/bioe.23041>
- Manzoni C., Kia D. A., Vandrovcova J., Hardy J., Wood N., Lewis P., and Ferrari R. 2018. Genome, transcriptome and proteome: the rise of omics data and their integration in biomedical sciences, *Briefings in Bioinformatics*, 19-2:286-302.
<https://doi.org/10.1093/bib/bbw114>
- Mathé C., Sagot M., Schiex T., and Rouzé P. 2002. Current methods of gene prediction, their strengths and weaknesses. *Nucleic Acids Research* 19:4103-4117.

- Montaseri S., Zare-Mirakabad F., and Moghadam-Charkari N. 2014. RNA-RNA interaction prediction using genetic algorithm. *Algorithms for Molecular Biology* 9:17. <http://www.almob.org/content/9/1/17>
- Moore J. H., Hahn L. W., Ritchie M. D., Thornton, T. A. and White, B. C. 2004. Routine discovery of complex genetic models using genetic algorithms. *Applied Soft Computing* 4:79-86. doi:10.1016/j.asoc.2003.08.003.
- Nizam A., Ravi J., and Subbaraya K. 2011. Cyclic genetic algorithm for multiple sequence alignment. *International Journal of Research and Reviews in Electrical and Computer Engineering* ISSN: 2046-5149.
- Notredame C., O'Brien E. A., and Higgins D. G. 1997. RAGA: RNA sequence alignment by genetic algorithm. *Nucleic Acids Research* 25:4570-4580.
- Oliveto P. S., and Witt C. 2014. On the runtime analysis of the Simple Genetic Algorithm. *Theoretical Computer Science* 545:2-19. ISSN 0304-3975. <https://doi.org/10.1016/j.tcs.2013.06.015>.
- Ooi C.H., and Tan, P. 2003. Genetic algorithms applied to multi-class prediction for the analysis of gene expression data. *Bioinformatics* 19:37-44.
- Pedersen J. T. and Moult J. 1997. Protein folding simulations with genetic algorithms and a detailed molecular description. *Journal of Molecular Biology* 269:240-259.

- Rashid M. A., Newton M. A. H., Hoque Md. T., and Sattar A. 2013. Mixing energy models in genetic algorithms for on-lattice protein structure prediction. *BioMed Research International* Volume 2013. <http://dx.doi.org/10.1155/2013/924137>.
- Reuter J and Mathews H. 2010. RNAstructure: software for RNA secondary structure prediction and analysis. *BMC Bioinformatics* 11:129.
<http://www.biomedcentral.com/1471-2105/11/129>
- Rogers A. and Prugel-Bennett A. 1999. Genetic drift in genetic algorithm selection schemes. *IEEE Transactions on Evolutionary Computation*, 3-4:298-303, Nov. 1999. doi: 10.1109/4235.797972
- Roth F. P., Hughes J. D., Estep P. W., and Church G. M. 1998. Finding DNA regulatory motifs within unaligned noncoding sequences clustered by whole-genome mRNA quantitation. *Nat Biotechnol.* 16(10):939-45.
- Ruffalo M., LaFramboise T., and Koyutürk M. 2011. Comparative analysis of algorithms for next-generation sequencing read alignment. *Bioinformatics* 27-20:2790-2796.
<https://doi.org/10.1093/bioinformatics/btr477>
- Havgaard J. H. and Gorodkin J. 2014. RNA structural alignments, part I: Sankoff-based approaches for structural alignments. *Methods Mol Biol.* 1097:275-90. doi: 10.1007/978-1-62703-709-9_13.

Sastry, K., and Goldberg, D.E. (2002). Genetic algorithms, efficiency enhancement, and deciding well with differing fitness variances. *Proceedings of the Genetic and Evolutionary Computation Conference*, 528–535. (Also IlliGAL Report No. 2002002).

Sîrbu A., Ruskin, H. J., and Crane M. 2010. Comparison of evolutionary algorithms in gene regulatory network model inference. *BMC Bioinformatics* 11:59.
<http://www.biomedcentral.com/1471-2105/11/59>

Spears, William M. 1995. Adapting crossover in evolutionary algorithms. Retrieved from: https://www.researchgate.net/publication/2263702_Adapting_Crossover_in_Evolutionary_Algorithms

Spears, W. M. and De Jong K. A. 1990. Using genetic algorithms for supervised concept learning. *Proceedings of the 2nd International IEEE Conference on Tools for Artificial Intelligence*. pp. 335-341. Herndon, VA, USA. doi: 10.1109/TAI.1990.130359

Stanke M. and Morgenstern B. 2005. AUGUSTUS: a web server for gene prediction in eukaryotes that allows user-defined constraints. *Nucleic Acids Research*, 33(Web Server issue), 465–467. <http://doi.org/10.1093/nar/gki458>

Tahmasebipour K. and Houghten S. 2015. Disease-gene association using a genetic algorithm. Conference paper presented at 2015 *IEEE Conference on*

Computational Intelligence in Bioinformatics and Computational Biology,
held at Niagara Falls, Canada in August 2015. doi:10.1109/CIBCB.2015.7300331

Taneda A. 2008. An efficient genetic algorithm for structural RNA pairwise alignment and its application to non-coding RNA discovery in yeast. *BMC Bioinformatics* 9:521. doi: 10.1186/1471-2105-9-521

Thompson J. and Gopal S. 2006. Genetic algorithm learning as a robust approach to RNA editing site prediction. *BMC Bioinformatics* 7:145. doi:10.1186/1471-2105-7-145

Thompson J. D., Higgins D. G., and Gibson T. J. 1994. CLUSTAL W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Research*, 22(22), 4673–4680.

Umarov R. K. and Solovyev V. V. 2017. Recognition of prokaryotic and eukaryotic promoters using convolutional deep learning neural networks. *PLoS ONE*, 12(2): e0171410. <http://doi.org/10.1371/journal.pone.0171410>

Xie X., Wu S., Lam K., and Yan H. 2006. PromoterExplorer: an effective promoter identification method based on the AdaBoost algorithm. *Bioinformatics* 22:2722-2728. doi:10.1093/bioinformatics/btl482

- Wall D. P., Pivovarov R., Tong M., Jung J.-Y., Fusaro V. A., DeLuca T. F., and Tonellato P. J. 2010. Genotator: A disease-agnostic tool for genetic annotation of disease. *BMC Medical Genomics*, 3, 50. <http://doi.org/10.1186/1755-8794-3-50>
- Winters-Hilt, S. and Merat, S. 2007. SVM clustering. *Proceedings of Fourth Annual MCBIOS Conference. BMC Bioinformatics* 8(Suppl. 7):S18. doi:10.1186/1471-2105-8-S7-S18
- Wu G., Pedrycz W., Li H., Qiu D., Ma M., and Liu J. 2013. Complexity Reduction in the Use of Evolutionary Algorithms to Function Optimization: A Variable Reduction Strategy. *The Scientific World* vol. 2013, Article ID 172193. <https://doi.org/10.1155/2013/172193>.
- Yang C., Moi S., Lin Y., and Chuang L. 2016. Genetic algorithm combined with a local search method for identifying susceptibility genes. *JAISCR* 6:203-212. doi:10.1515/jaiscr-2016-0015
- Zara-Mirakabad F., Ahrabian H., Sadeghi M., Hashemifar S., Nowzari-Dalini A., and Goliaei B. 2009. Genetic algorithm for dyad pattern finding in DNA sequences. *Genes and Genetic Systems* 84: 81-93.