BIOINFORMATICS ORIGINAL PAPER

Vol. 25 no. 13 2009, pages 1640–1646
doi:10.1093/bioinformatics/btp283

Gene expression

Optimizing static thermodynamic models of transcriptional regulation

Denis C. Bauer* and Timothy L. Bailey*

Institute for Molecular Bioscience, The University of Queensland, Brisbane, Qld. 4072, Australia

Received on January 22, 2009; revised on March 31, 2009; accepted on April 21, 2009

Advance Access publication April 27, 2009

Associate Editor: Olga Troyanskaya

ABSTRACT

Motivation: Modeling transcriptional regulation using thermodynamic modeling approaches has become increasingly relevant as a way to gain a detailed understanding of transcriptional regulation. Thermodynamic models are able to model the interactions between transcription factors (TFs) and DNA that lead to a specific transcriptional output of the target gene. Such models can be 'trained' by fitting their free parameters to data on the transcription rate of a gene and the concentrations of its regulating factors. However, the parameter fitting process is computationally very expensive and this limits the number of alternative types of model that can be explored.

Results: In this study, we evaluate the 'optimization landscape' of a class of static, quantitative models of regulation and explore the efficiency of a range of optimization methods. We evaluate eight optimization methods: two variants of simulated annealing (SA), four variants of gradient descent (GD), a hybrid SA/GD algorithm and a genetic algorithm. We show that the optimization landscape has numerous local optima, resulting in poor performance for the GD methods. SA with a simple geometric cooling schedule performs best among all tested methods. In particular, we see no advantage to using the more sophisticated 'LAM' cooling schedule. Overall, a good approximate solution is achievable in minutes using SA with a simple cooling schedule.

Contact: d.bauer@uq.edu.au; t.bailey@imb.uq.edu.au

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 INTRODUCTION

Transcription of a gene can be induced by the binding of specific transcription factor (TFs) proteins to genomic regions called *cis*-regulatory modules (CRMs). Bound TFs can act to either enhance or repress transcription of the target gene, and this action can depend on the relative locations within the CRM where the TFs are bound. Bound TFs can also interact with each other, and the full range of effects of such interactions on transcription is not well understood at present. One hypothesis is that some TFs can act as repressors of transcription by negating the effect of activating TFs bound within a *limited distance* away within the CRM (Reinitz *et al.*, 2003).

The frequency and duration of the binding events of TFs to a CRM are influenced by thermodynamic properties of the TF protein molecules and the DNA of the CRM. The probability of a TF being bound to a particular site within the CRM is a function of its concentration and its binding affinity to the particular sequence of nucleotides making up the site. This affinity can be modeled as a function of the log-odds score of the binding site, the ratio of the probability of the site under a position-specific model to the 'background' probability of the site (Stormo, 1998), and can be captured by a position-specific weight matrix (PWM) motif for the specific TF. Knowing the PWM for a TF allows standard equations for reversible binding events to be used to estimate the probability of the bound and unbound state at a given site in the CRM. These equations can also be extended to account for competition for sites by TFs with similar binding affinity profiles, as well as competition for overlapping sites by molecules of a single type of TF.

These ideas have been combined recently to produce several variations of static, thermodynamic models of transcription (Bauer and Bailey, 2008b; Janssens *et al.*, 2006; Segal *et al.*, 2008; Zinzen and Papatsenko, 2007). These models allow us to predict *in silico* the transcription rate of the modeled gene, given the DNA sequence of its CRM and the concentrations of each of the TFs that affect its transcription. Such models have many uses. They allow us to propose testable hypotheses regarding the effect of mutations in the CRM on the transcription rate of the gene. By building and validating the predictions of additional variations of these models, hypotheses regarding different modes of interaction among the TFs regulating a gene can be examined *in silico*. Failure of a model to accurately account for the response of a gene can suggest that the model is missing inputs (e.g. missing regulatory TFs).

Thermodynamic models of transcription contain a number of free parameters. Typically, the models have up to two free parameters per TF in addition to one or more model-wide parameters. The model proposed by Reinitz *et al.* (2003) (herein called the 'Reinitz' model), contains two parameters per TF, one of which is proportional to the maximum association constant of the TF-DNA binding and the other quantifies the 'effectiveness' of the TF as either an activator or a repressor. (The role of the TF can be specified by the user, but could also be considered a binary free parameter.) The Reinitz model contains two additional parameters representing the maximal transcription rate and the Gibbs free energy barrier to transcription. The model proposed by Segal *et al.* (2008) additionally includes one parameter per TF, describing the strength of synergistic effects, as well as the PWMs as free parameters, greatly increasing the total number of free parameters.

Creating a thermodynamic model of transcription requires tuning the free parameters to minimize the difference in predicted

*To whom correspondence should be addressed.

and observed rates of transcription. This amounts to function optimization, where the inputs to the function are measurements of TF concentrations and the output is the transcription rate of the modeled gene. Because the models are highly non-linear and have many free parameters, their optimization requires large amounts of computer time. As a result, the computational cost of creating a model limits our ability to explore refinements and variations that would shed additional light on the mechanisms of regulation.

The motivation of the current work is to reduce this bottleneck in the studies of static, thermodynamic models of regulation. To this end, we study the relative effectiveness of a number of standard function optimization methods when training such models. To isolate the optimization problem from the fact that the models are only an approximation of reality, we train using synthetic data that is very similar to—and derived from—real, biological data. To learn about the difficulty of the problem, we first compare the speed and accuracy of the standard optimization methods and devise a number of variants tailored to this particular problem. We then examine the optimization 'landscape' in detail. We evaluate members of three optimization method categories: simulated annealing (SA), gradient descent (GD) and genetic algorithms (GA). We also test a hybrid SA-GD algorithm, which combines global and local search approaches. All experiments are performed using STREAM, our publicly available framework for modeling target-specific transcriptional regulation (Bauer and Bailey, 2008a).

2 METHODS

2.1 Static, thermodynamic models of transcription

The experiments reported in this article use three variations of the Reinitz model. We choose the Reinitz model because it is based on fundamental biological concepts and has been studied well in previous works (Bauer and Bailey, 2008b; Janssens *et al.*, 2006). Furthermore, it contains far fewer free parameters than other methods (e.g. Segal *et al.*, 2008) and is thus quicker to train. Both types of model are based on very similar assumed mechanisms of regulation. We therefore believe that the observations made in this study about the optimization landscape and performance of the optimization methods, should largely be applicable to the other thermodynamic model approaches (Segal *et al.*, 2008; Zinzen and Papatsenko, 2007).

The Reinitz model has been described in detail elsewhere (Janssens *et al.*, 2006). In a nutshell, it models the transcription rate of a gene as a function of a vector of model-free parameters, Θ, and a vector of model inputs, $\mathbf{X}$. The free parameters comprise a maximum association constant (K) for each TF, an 'effectiveness' constant for each TF, (E), the Gibbs free energy threshold (G_0) of transcription and the maximal transcription rate (R_0). The role—activator or repressor—of each TF is specified by the user, and not treated as free. The model inputs are the concentrations of a set of TFs, and a set of positions and log-odds scores for (high-scoring) TF binding sites (TFBSs) in the controlling CRM. (We call this set of positions and log-odds scores a 'TFBS-map'.)

The transcription rate function computed by the Reinitz model is

$$R(\mathbf{X},\Theta) = \begin{cases} R_0 \exp(M(\mathbf{X},\Theta)-G_0) & \text{if } M(\mathbf{X},\Theta) < G_0 \\ R_0 & \text{otherwise,} \end{cases} \tag{1}$$

where $M(\mathbf{X},\Theta)$ is the decrease in free energy caused by the 'effective' number of bound activators. The function $M(\mathbf{X},\Theta)$ is differentiable, and it takes into account the positions and log-odds scores of sites and the concentration of each of the TFs. It also incorporates competition for sites by TFs and the reduction in activation caused by bound repressors, which 'quench' nearby bound activators. Details are given in Janssens *et al.* (2006).

The function in Equation (1) presents two difficulties for optimization. First, all of the free parameters must be non-negative in order to be biologically meaningful, so the optimization must be *constrained*. (It makes no sense, for instance, to talk about negative TF-DNA association constants.) Second, the output is 'capped' (at a maximum value of R_0) so the function is not continuously differentiable.

In order to easily use gradient information in the optimization of the transcription rate function, the objective function should be continuously differentiable. We therefore test two additional variants of Equation (1) with GD. We call the original Reinitz function 'max'. One variant, 'nomax', simply removes the cap at a maximum transcription rate of R_0. The second variant, which we call 'softmax', uses the sigmoid function, $\mathrm{sigm}(x)$, to cap the rate function at R_0.

The three variants of the Reinitz rate function can all be expressed as

$$R(\mathbf{X},\Theta) = R_0 F(g), \tag{2}$$

where $g = M(\mathbf{X},\Theta) - G_0$, and $F(g)$ has one of the three forms

$$\text{max}: F(g) = \begin{cases} \exp(g) & \text{if } g<0, \\ 1 & \text{otherwise,} \end{cases}$$

$$\text{softmax}: F(g) = \mathrm{sigm}(g)$$

$$\text{nomax}: F(g) = \exp(g).$$

The behaviors of these three versions of $F(g)$ is illustrated in the Supplementary Section 1, Paragraph 3. The sigmoid function used in the 'nomax' variant is defined as

$$\mathrm{sigm}(g) = \frac{1}{(1+\exp(-(g\cdot b)-c))},$$

with $b = 2.5$ and $c = 2$. These values of b and c were chosen to shift and steepen $F(g)$ to achieve a closer resemblance of 'softmax' to 'max' for legal values of x. The sigmoid transformation causes the function to be smooth and differentiable.

When using unconstrained optimization methods like GD, we rewrite Equation (2) to use transformed parameters. Optimization is done in the unconstrained, transformed space, and the inverse transformation is used to retrieve the underlying parameter values in the original space of the rate function. Our parameter transformation limits the actual value of the parameters to a user-defined upper limit and to a lower limit of zero. (Details of the parameter transformations and ranges are given in the Supplementary Sections 1 and 5.)

2.2 Optimization methods

We use optimization methods to attempt to find values for the model-free parameters that minimize the root mean-squared (RMS) error, B, between the *known* transcription rate and the rate *predicted* by the model, averaged over a set of input points (Bauer and Bailey, 2008b). The RMS error function $B(D,\Theta)$ takes a set of input data points, D (TF expression, TFBS-map, known transcription rate), as well as a specific set of parameter values, Θ, and calculates the RMS error. Where it is clear what input data points, D, we are referring to, we abbreviate the notation to $B(\Theta)$.

Our goal is to determine which function optimization method is fastest (in terms of computer time) at training static, thermodynamic transcription models. The function optimization methods we study here are variants of SA, GD with random restart and GA. Whereas SA and GA are global optimization strategies, GD is a local function optimizer. We therefore combine GD with random restart from a different set of initial parameters. The hybrid SA-GD algorithm we test uses GD to locally refine a single solution proposed by SA in the hope of faster convergence.

2.2.1 Simulated annealing The SA global function optimization method is an iterative method that starts with an initial (random) estimate of the function parameters, $\Theta^{(0)}$, and computes its prediction error, $B(\Theta^{(0)})$. At each succeeding iteration, i, SA randomly selects a value of Θ in the region of parameter space near the current estimate of the parameters, $\Theta^{(i)}$, and accepts Θ as the new estimate, $\Theta^{(i+1)}$, if it has lower error than the current estimate. If Θ has larger error it may still be accepted depending on the size of the difference in error and the 'cooling schedule'.

An SA method has been used by us and others previously for optimizing Reinitz models of transcription in *Drosophila* (Bauer and Bailey, 2008b; Janssens *et al.*, 2006). The SA variant in those studies was developed by Chu *et al.* (1999), and uses the so-called LAM cooling schedule. The LAM cooling schedule takes the properties of the problem into account when calculating the exponential cooling function (Lam, 1988). The optimization will stop if the system is considered 'frozen' for a specified number of iterations, z, where 'frozen' means the temperature has dropped below a defined value κ. The values of z and κ affect the number of optimization steps and therefore the computer time consumed. We will refer to this optimization method as 'SA_LAM'.

The second SA variant we examine uses a simple, geometric cooling schedule. This method sets the 'temperature' for iteration i as

$$T^{(i+1)} = r^{(i)} T^{(i)},$$

where the reduction factor, $r^{(i)}$, is defined as $r^{(i)} = \exp\big(\log(0.006)/i\big)$, and r is adjusted with the user determined maximal number of iterations, n, to keep it at a value larger than zero at the n-th iteration (here 0.006). This method stops examining new potential solutions when a user-specified maximum number of iterations, n, has been reached, or when the RMS error is $< 10^{-6}$. We refer to this optimization method as 'SA_geom'.

SA uses a so-called 'neighborhood' function to propose new candidate values, $x^{(i+1)}, \forall x \in \Theta$. The neighborhood function randomly samples from a ball around the current parameter x with a radius, p, that is maximally 30% of the parameter range and that shrinks with decreasing temperature.

Our neighborhood function chooses a new value for each parameter $x \in \Theta$ by first determining whether to decrease or increase x by randomly and uniformly choosing $W \in \{-1, 1\}$, then choosing the size of the change, $\delta(x^{(i)})$, then multiplying by the current temperature, $T^{(i)} \leq 1$ and finally adding the result to the current value of x:

$$x^{(i+1)} = x^{(i)} + W\delta(x^{(i)})T^{(i)}.$$

The neighborhood function chooses the size of the change, $\delta(x^{(i)})$, by sampling uniformly from the interval $[0, p]$, where the sampling radius, p, is

$$p = \begin{cases} \min(0.3R_x, x^{(i)} - L_x) & \text{if } W = -1 \\ \min(0.3R_x, U_x - x^{(i)}) & \text{if } W = 1. \end{cases}$$

The definition of p insures that the new estimate of x remains within its legal interval $[L_x, U_x]$, and R_x is defined as $R_x = U_x - L_x$. We limit the valid ranges to be biologically plausible when generating a new parameter set by the neighborhood function (see Supplementary Section 5). See Supplementary Section 6 for a comparison of various neighborhood functions.

2.2.2 Gradient descent We also study random-restart GD to optimize the transcription rate model. This method starts from a randomly chosen value of the free parameters, Θ, and follows the gradient of the RMS error function, until the improvement in RMS error between one iteration and the next is smaller than the user-defined value ϵ_c. Since convergence may be at a local minimum, the above process is repeated from a new random value of Θ. The method stops evaluating new solutions when the user-defined maximal number of iterations, n, has been reached.

As noted above, the transcription rate function in Equation (1) is not continuously differentiable, so neither is the RMS error function derived from it. However, it is differentiable when $M \neq G_0$. Two of our variants of Equation 2, 'softmax' and 'nomax', are differentiable, as are the derived error functions. The partial derivatives with respect to each parameter in Θ of the error functions are given in Supplementary Section 1, Paragraph 2. We will refer to the GD applied to these three models as 'GD_max', 'GD_softmax' and 'GD_nomax', respectively.

The GD methods minimize $B(\Theta)$ using the update rule

$$x^{(i+1)} = x^{(i)} - \left(\nu \cdot \frac{\partial B(\Theta^{(i)})}{\partial x}\right), \forall x \in \Theta, \tag{3}$$

where $\Theta^{(i)}$ is the current values of the free parameters, $\Theta^{(i+1)}$ is the *proposed* set of updated values of the parameters, and ν is the 'learning rate'. If the change in all parameters is $< 10^{-16}$, the algorithm stops. If the proposed solution is legal (all parameters are within their legal ranges), and its error is less than that of the previous solution, it is accepted, and $\Theta^{(i+1)}$ replaces $\Theta^{(i)}$, and ν is *increased* by 5%. Otherwise, $\Theta^{(i+1)}$ is rejected, ν is *reduced* by 90% and the update rule is reapplied to the current solution ($\Theta^{(i)}$). When the reduction in error between two accepted solutions is less than ϵ_c, the current solution is saved if it is the best so far, the iteration budget is reduced by i, and the method restarts with a new, random value of $\Theta^{(0)}$. Unless otherwise noted, we set ϵ_c to 10^{-6}.

When optimizing the 'max' version of the model, a proposed solution is rejected if it predicts a value of R larger than R_0 for any point in the input data. If this is the case, ν is reduced by 90% and the update rule is reapplied to the current solution. In this way, the optimization can find a gradient leading away from the invalid solution space or converging to a good solution very close to the boundary.

Since the RMS error function seems to have large flat regions that make GD inefficient, we also study a variation of the above GD optimization method motivated by 'resilient back-propagation' (Rprop) (Riedmiller and Braun, 1994). This version of GD, which we call 'GD_Rprop', uses only the sign of the gradient in its update rule and can be applied to all GD variants, replacing the derivative in Equation 3 with

$$\operatorname{sign}\left(\frac{\partial B}{\partial x}\right) = \begin{cases} 1 & \text{if } \frac{\partial B}{\partial x} > 0, \\ -1 & \text{if } \frac{\partial B}{\partial x} < 0, \\ 0 & \text{otherwise}, \end{cases} \tag{4}$$

for all $x \in \Theta$.

2.2.3 SA-GD hybrid algorithm We also test a hybrid method that uses the SA_geom first to perform a global search before switching to GD_max and following the gradient to the nearest optimum. The hybrid splits the given iteration budget and gives two-thirds to the SA_geom and one-third to the GD method. Only one switch is made from SA to GD. Other splits have been tested but produce inferior results (data not shown).

2.2.4 Genetic algorithms A function optimization approach that is more directed than SA, and less technically constrained than GD, is GAs (Holland, 1975). Rather than exploring the whole parameter space, GAs restrict the search to the neighborhood of successful solutions. The search is inspired by the biological processes of mutation and recombination, which introduce small changes to an already successful solution. The objective of GA is to constrain the search space to promising regions.

We use the JGAP (http://jgap.sourceforge.net) java-implementation of the GA to optimize the RMS error of the transcription rate model. Each of the free parameters of the model is defined to be a 'gene', and concatenated to collectively represent Θ as a 'chromosome'. Each gene can be constrained to take only values from a defined value range (see Supplementary Section 5). We generate 80 different chromosomes representing 80 different Θ. For each chromosome, we assess its fitness as the RMS, $B(\Theta)$. In the subsequent 'selection round', only the fittest chromosome is preserved all others undergo mutation and recombination steps and the whole population enters the next selection round. The number of selection rounds can be varied in order to control the total computer time consumed.

2.3 Experimental data

2.3.1 Biological data We use two experimental datasets from *Drosophila* blastocysts (Janssens *et al.*, 2006; Segal *et al.*, 2008). The first dataset, 'eve_stripe2_multitime', contains 406 measurements of mRNA levels of a gene construct containing the *eve* gene MSE2 CRM (Janssens *et al.*, 2006). Each measurement corresponds to a different time point during development and a different position along the antereoposterior axis of a fruit fly embryo. The second dataset, 'dros_singletime', contains mRNA levels of 10 different fruit fly development gene CRM constructs (Segal *et al.*, 2008) measured at 100 different anteroposterior positions. Unlike the eve_stripe2_multitime data, the responses are only measured in binary levels (on/off) and only at one time point. We chose 10 CRMs randomly from the 35 for which Segal *et al.* (2008)'s model achieved a 'good' (18) or 'fair' (17) fit. Seven of the 10 CRMs were judged to have a 'good' fit—'d_+4', 'eve_stripe_46', 'hb_anterior' 'Kr_CD1_run', 'pdm2', 'run_stripe3' and 'run_stripe5'—and three to have a 'fair' fit—'cad_+14', 'h_stripe34' and 'slp_-3'. We chose to include fair-fitting CRMs to simulate the level of noise that has to be expected in real data.

Both datasets also contain concentration data, measured at the corresponding time and positions along the embryo, for the TFs Bicoid, Caudal and Hunchback as activators, and Giant, Knirps, Krüppel and Tailless as repressors (Janssens *et al.*, 2006). For the second dataset, following Segal *et al.* (2008), we designate Hunchback to be a repressor and we include an additional activator—Torso-response element.

The eve_stripe2_multitime dataset, a single TFBS-map with the positions and log-odds scores of *in silico* predicted TFBSs in the 1.7 kb DNA region upstream of the *eve* gene, as described in Bauer and Bailey (2008b). Only sites with predicted log-odds scores of at least 9.0 bits are included in the TFBS-map. The dros_singletime dataset contains a separate TFBS-map for each of the 10 CRMs created in the same way.

2.3.2 Synthetic data The definitive quality of the found solutions generated by the optimization methods, can only be evaluated if the 'correct' solution is known. Since we do not know what the correct solution of the biological data is, we have no means to quantify the distance of the found to the correct solution.

We derive a set of synthetic data from eve_stripe2_multitime data, to obtain training data for which we know the optimal value of the parameters, and for which the optimal RMS error is zero. We use GD_nomax to find a value of Θ that (hopefully) is close to the correct solution, which we call $\hat{\Theta}$. We then replace each of the observed transcription rate values, R, with values $\hat{R}=R(\mathbf{X},\hat{\Theta})$, which are the values predicted by the model with Θ set to $\hat{\Theta}$.

GD_nomax was chosen over GD_softmax or GD_max because it uses the original Reinitz objective function and is therefore less biased towards one of the variants. Whether GD or SA is used to optimize this objective function does not influence the nature of the synthetic data (data not shown). GD_nomax was allowed a total iteration budget of 5000 and trained on all the data. The synthetic data were used in all experiments unless stated otherwise. A comparison between the real and 'corrected' data can be seen in the Supplementary Section 9.

2.4 Evaluating the optimization methods

Using the synthetic data described above, we measure the average, cross-validated accuracy of the models found by the different optimization methods as a function of the amount of computer time required. Our accuracy metric is Pearson's correlation coefficient (CC). The CC between the prediction and the synthetic transcriptional output provides a normalized estimate (between -1 and 1, with one being the maximal correlation) of the quality of the trained model. We chose to use the CC to evaluate the trained models because it measures whether the 'shape' of the model predictions matches the 'shape' of the 'correct' transcriptional output, and compared with the RMS error metric, CC is less sensitive to discrepancies in the magnitude of the predicted output.

We run each optimization method with a series of time budgets ranging from 1 to 400 min on a single x86_64 processor under Linux. For each method and budget, we perform 5-fold cross-validation, training on four-fifths of the data points and measuring accuracy on the remaining one-fifth. We do this in order to estimate the true generalization accuracy achieved in the given time. We repeat each cross-validation experiment 15 times using different seeds, which generates a different cross-validation split, as well as different parameter values for each restart of the methods. We report the average CC over the cross-validation splits and restarts.

2.5 Exploring the optimization landscape

Using the synthetic data described in the previous section, we analyze the 'basin of attraction' of the global minimum. We define the basin of attraction around the known solution, $\hat{\Theta}$, as all initial values, Θ, from which GD converges back to $\hat{\Theta}$. We explore the basin of attraction in two ways. First, we examine the region around $\hat{\Theta}$ by perturbing one dimension at a time. Second, we explore the entire landscape, by sampling random values of Θ.

We calculate the *scaled* distance in parameter space between a solution found by GD, Θ', and the known solution $\hat{\Theta}$, as

$$D(\Theta',\hat{\Theta}) = \sqrt{\frac{1}{n}\sum_{x\in\Theta}\left(\frac{\hat{x}}{U_x}-\frac{x'}{U_x}\right)^2},$$

where U_x is the maximal value the parameter x is allowed to have. The normalization of each parameter value to range between zero and one insures that all parameters have the same impact on the sum of changes in D. For comparison, we similarly compute the scaled distance between a starting point for GD and the known solution, $D(\Theta^{(0)},\hat{\Theta})$.

Our first study examines the landscape in the immediate neighborhood of the known solution, $\hat{\Theta}$. To determine the basin of attraction of $\hat{\Theta}$, we perform the following steps. First, we perturb one parameter of $\hat{\Theta}$ by multiplying the parameter value by either 0.99 or 1.01 to generate a starting point, $\Theta^{(0)}$, for GD. Second, we run GD_nomax from $\Theta^{(0)}$ for $n=10000$ iterations or until the change in RMS error is below the machine precision ($\epsilon_c=10^{-16}$) to identify the nearest minimum, Θ'. This is repeated independently for each parameter. (Note that GD is not allowed to restart in these experiments.)

In our study of the basin of attraction of $\hat{\Theta}$, we execute GD_nomax in two different ways. First, we prevent all parameters except the perturbed one from being updated by GD. This ensures that Θ moves only in the dimension of the perturbation and all other parameters are held fixed at their optimal values. Second, GD is allowed to adjust all parameters. The first experimental setup tests the 'smoothness' of the landscape in only one dimension, while the second tests the convergence in all dimensions.

Our second study examines the global landscape. We randomly sample m (legal) values of $\Theta^{(0)}$ and observe the convergence of GD_nomax. We run GD_nomax for $n=10000$ iterations or until the change in RMS error is below ($\epsilon_c=10^{-16}$) to identify the nearest minimum, Θ'.

3 RESULTS

3.1 Comparison of optimization methods

In this section, we compare the optimization methods on the biological datasets using cross-validated Pearson's CC. The rankings of the methods are unchanged using cross-validated RMS error as the evaluation criterion (See Supplementary Section 10).

3.1.1 Simulated annealing We first examine the performance of variants of the SA algorithm. There seems to be little benefit in using the more sophisticated LAM cooling schedule with the SA algorithm (SA_LAM), as illustrated in Figure 1. Our SA-variant with a simple geometric cooling schedule (SA_geom) performs equally well if not better with all time budgets.

Our SA algorithm with geometric cooling also finds reasonable solutions in a very short time. For example, with this data, models found by SA_geom after only 60 s of run time have an average, cross-validated CC of 0.97. An additional advantage of SA_geom is that its cooling schedule can be adjusted to terminate after any desired amount of run time, whereas, due to technical issues, the SA_LAM cooling schedule always requires at least 16 min with this problem (hence the missing points in Fig. 1).

Nonetheless, the convergence of the SA algorithms is slow after the first minute of run time, with both SA-based optimization

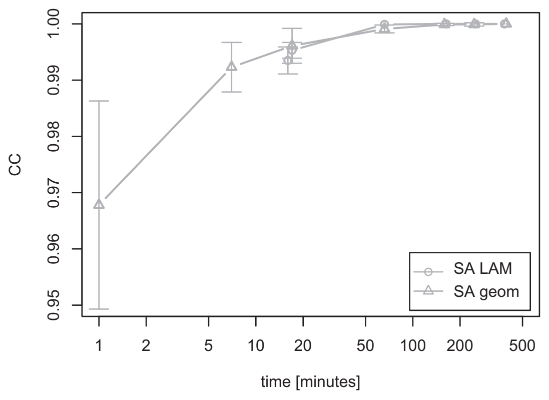

Fig. 1. Effectiveness of SA with different cooling schedules. The figure shows the cross-validated CC as a function of run time (log scale) for the two different SA optimization methods.

methods requiring about 50 min to find models with average CC very close to 1.0. This illustrates the high-computational cost—∼1 h—of training a single static, quantitative model of transcription of this size (7 TFs, 20 binding sites) with this amount of data (406 points, 20 TFBSs) using SA-based algorithms. A comparison of the real and predicted transcriptional output with CC = 0.97 and 1.0, respectively, is given in the Supplementary Section 4, which illustrate the usefulness of the model trained for 1 min only.

3.1.2 Gradient descent We hoped to reduce the high-computation cost of the SA-based algorithms by using GD-based approaches, which take advantage of knowledge of the derivative of the error function. Surprisingly, compared with the SA_geom method, all three GD-based methods find inferior models for all run times up to about 200 min on the MSE2 dataset (Fig. 2a). The superiority of SA_geom is even clearer when 10 CRMs are fit simultaneously (dros_singletime dataset) where, even after more than 200 min, the GD-based methods find models with <70% of the accuracy (CC) found by models using SA_geom (Fig. 2b).

We tried another GD-based variant designed to reduce the effect of the optimization landscape. This method, GD_Rprop, uses only the sign of the gradient. Since it uses a gradient-independent step size it can proceed faster through 'plains' (regions with low gradient) and might not 'step over' a minimum if the slope leading to it was steep. GD_Rprop was able to increase the performance for some GD variants for short runs but could not improve beyond the performance of GD_nomax (Supplementary Section 2).

3.1.3 SA-GD hybrid Our hybrid SA-GD optimization method does not perform substantially better than the SA_geom method, as seen in Figure 3. With very short running times, the average CC of the models found by the hybrid method is slightly higher than those found by SA_geom, but this difference is not statistically significant (paired *t*-test). For running times longer than 5 min, the quality of the solutions found on this data by the two methods are indistinguishable. Note that the SA-GD hybrid uses our GD_max local optimization method, which shows inferior performance to the GD_nomax method when the time budget is small (Fig. 2). However, the GD_nomax optimizes a slightly different model than SA_geom, so we did not construct an SA_geom-GD_nomax hybrid. Note, also, that our hybrid runs GD from only a single solution found by SA, and we did not test other possible hybrid strategies.

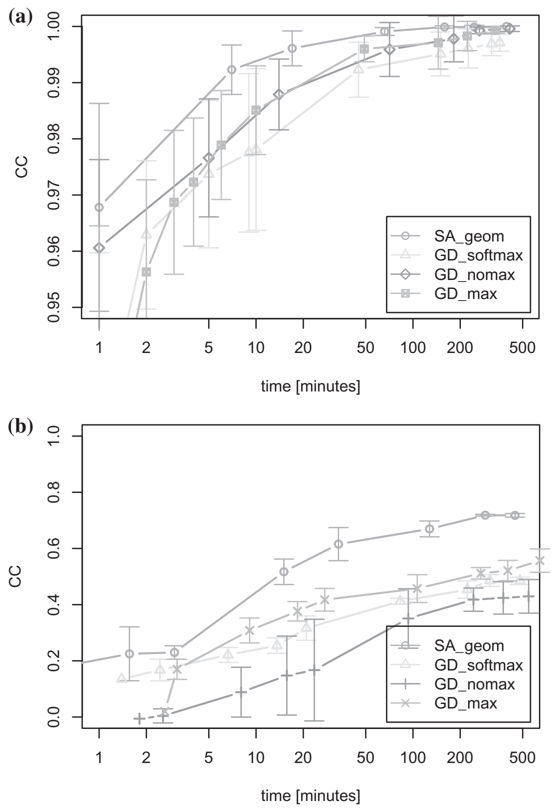

Fig. 2. Effectiveness of variants of GD. The figure shows the cross-validated CC as a function of run time (log scale) for the three GD variants compared with the SA_geom optimization method on the eve_stripe2_multitime dataset (**a**) and the dros_singletime dataset (**b**).

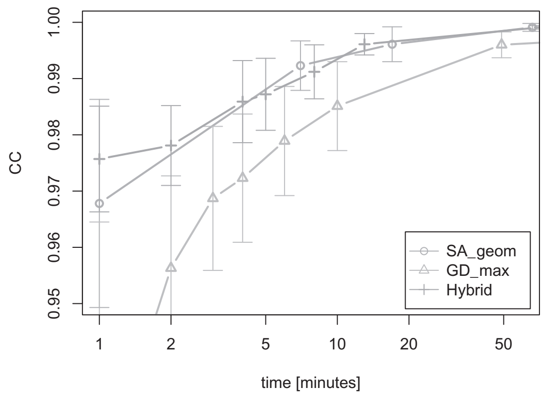

Fig. 3. Effectiveness of SA-GD hybrid optimization algorithm. The figure shows the cross-validated CC as a function of run time (log scale) for the SA-GD hybrid, GD_max and SA_geom optimization methods.

3.1.4 Genetic algorithm The GA optimization method performs the worst among all the methods we tried (Supplementary Section 8). Not even after an extensive 700 min training time was the GA

Table 1. Convergence behavior of GD.

Experiment	$D(\Theta^{(0)},\hat{\Theta})$ (mean)	$D(\Theta',\hat{\Theta})$ (mean)	$\|\nabla B(\Theta')\|$ (mean)	Error Red. (%)
1% (held)	3.4×10^{-4}	0	1.4×10^{-13}	100
1% (free)	3.4×10^{-4}	2.8×10^{-4}	4.3×10^{-5}	88
Random	0.10	0.11	0.04	96.99

Column one: 1% was added or subtracted from a single parameter, or the parameter vector (Θ) was randomly sampled; 'held'—all parameters were held fixed at their optimal value during GD except the perturbed parameter; 'free'—all parameters are updated by GD. Column two: the initial distance to the known solution. Column three: the final distance to the known solution. Column four: the final gradient. Column five: the improvement (reduction) in the error function. All values are means for 32 sets (16 parameters, ±1% perturbation) or 100 sets (random sampling) using the GD_nomax algorithm.

method able to match the performance of any of the SA or GD variants. The variance in performance remains considerably large as shown by the SD bars.

3.2 The optimization landscape

We were somewhat surprised that the GD-based methods, are less efficient at optimizing the transcription rate RMS error function than SA. In order to understand why, we investigate the properties of the optimization landscape of the error function using *synthetic* data based on real, biological data. As described earlier, we conduct two studies. First, we determine the basin of attraction of the known global optimum in the optimization landscape of the Reinitz model. To do this, we perturb each parameter individually and examine the properties of Θ' at convergence of GD_nomax. Second, we study convergence from random points in the parameter space. The results of both studies are summarized in Table 1.

The GD algorithm (GD_nomax) always finds the known solution when started from a point where one parameter has been perturbed 1% *and* we constrain the algorithm only to search along the perturbed dimension. Under these conditions, the mean distance between the known and found solutions is exactly zero (row 1, column 3 of Table 1). This indicates that the landscape is smooth and reasonably 'steep' around the solution, when projected along one dimension (see Supplementary Section 3). It also shows that the optimization algorithm works correctly.

However, when our implementation of GD is started from a point near the known solution and allowed to change the parameters in all, rather than only in the perturbed dimension, it rapidly finds a nearby region of very low gradient, but not the known solution. A typical case is illustrated in Figure 4, which shows how the RMS error function, distance to the known solution and size of the gradient change during GD. The solutions found by GD are only marginally closer to the known solution than the 1% starting point, but have dramatically lower RMS error. As in this typical case, the size of the gradient approaches zero after only about 20 iterations. (Refer to the Supplementary Section 3 for similar plots for all model parameters.)

The generality of the convergence behavior illustrated in Figure 4 is made clear in Table 1, where row 2 shows that, on average, the distance to the known solution from the solution found by GD decreases only marginally—from 3.4×10^{-4} initially to 2.8×10^{-4} after (at most) 10 000 iterations. The table also shows that, on average, GD succeeds in reducing the error by 88%, and that, when GD stops, the norm of the gradient is very small—4.3×10^{-5}.

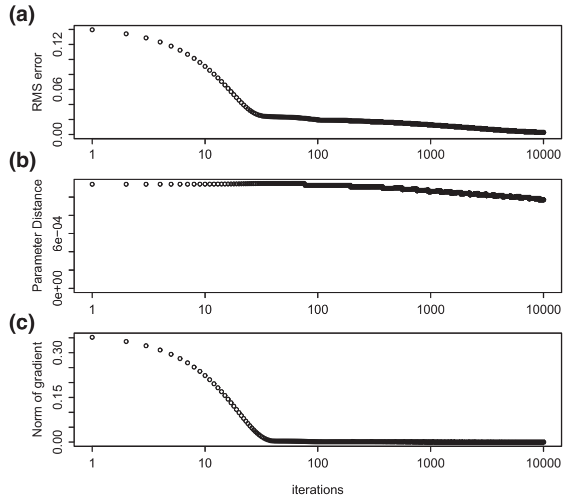

Fig. 4. Typical convergence behavior of GD near known optimal solution. The panels show, as function of iteration number, i: (**a**) RMS error, $B(\Theta^{(i)})$; (**b**) distance to the known solution, $D(\Theta^{(i)},\hat{\Theta})$; (**c**): size of the gradient, $\|\nabla B(\Theta^{(i)})\|$.

It is clear, therefore, that the region of the RMS error function space surrounding the known solution is extremely flat, which makes it difficult for GD to converge, and many have local minima. The best evidence for this is that, for two of the parameters in the 1% perturbation experiment, GD finds solutions that are more distant from the known solution than the initial starting point ($\Theta^{(0)}$) is (see Supplementary Section 3). In 14 out of 16 parameters, however, GD finds solutions that are (somewhat) closer to the known solution than the starting point is, as illustrated in Figure 4.

To further investigate the optimization landscape of the transcription rate error function, we study the behavior of GD_nomax when it is started from $m=100$ completely *random* starting points. We observe that GD *never* finds the global optimum from any of the 100 random starting points we try. On average, the solutions found are slightly further from the known solution than the starting points are (Table 1, row 3), and have 97% smaller error. However, the closest solution found is distance 0.04 from the optimal solution in (scaled) parameter space, and has error 0.61 (results are shown in Supplementary Section 7).

When started from a random point, GD gets trapped in two types of local minima. Referring again to the Section 7 of the Supplementary Material, we see that for 47 out of 100 random starts, GD converges to solutions that predict *zero* transcription. These solutions have extremely small gradients ($\|\nabla B(\Theta')\| < 10^{-8}$) and high errors ($B(\Theta')\approx 41$). In the other 53 runs, GD finds solutions with small gradients (mostly <0.05) and much smaller error (average 7.24, median 3.07). These local minima are not all the same, since they are at varying distances from the known solution.

In conclusion, the Reinitz function appears to be difficult to optimize using gradient-based methods. First, the region around the known optimal solution is extremely flat, causing extremely slow convergence near the optimum. Second, the number of local optima seems to be very large, causing the optimizer to get stuck or to move away from the known solution.

4 DISCUSSION

Our study of the static, quantitative models of transcription of the type described in Reinitz *et al.* (2003) ('Reinitz' models) show that they are difficult to optimize using gradient-based methods. This difficulty becomes especially prevalent when the amount of data to be fit simultaneously grows or the diversity within the data increases. The most effective optimization method in our study is SA with a simple geometric cooling schedule. Previous studies used SA with the sophisticated LAM cooling schedule, but we see no evidence that the more complicated algorithm is superior.

We have shown that Reinitz models have optimization landscapes with a very large number of local minima. This present obvious difficulties for optimization methods that use GD. Our experiments of comparing several GD-based methods with SA methods bear this out—using SA is superior to using the GD approaches presented here.

We also observe that the optimization landscape is very flat near the known optimum. This causes the GD algorithms to converge extremely slow. Although we did not thoroughly explore hybrid SA-GD algorithms, the large number of local minima and slow convergence of GD near the known solution indicates that there might not be a large improvement unless ways to overcome this slow convergence are found. If the convergence rate could be improved, hybrid algorithms that run GD from many solutions proposed by SA could be effective. This approach is called basin-hopping, and has proved useful for predicting protein folding (Prentiss *et al.*, 2008).

The GA we studied performs very poorly compared with SA and GD. The performance of GA might be improved by extensive parameter tuning, but this would increase the risk of adapting the optimization framework too much to the problem and therefore biasing the performance estimation.

Further progress in optimizing Reinitz transcription rate models using GD will require finding ways to reduce the number of local minima and to increase the convergence rate. Almost half of the local minima found in our experiments corresponded to models that predict no transcription. These models have the maximal transcription rate parameter, R_0, set close to zero. Simply constraining R_0 to larger values might help reduce the number of this type of minimum. Such 'trivial' minima tend to be found after very few iterations of GD, however, so they have less effect on efficiency than the non-trivial minima.

The non-trivial minima may be due to the form of the model, which includes two parameters for each transcription factor. Since one parameter determines the binding affinity and the other the effectiveness of the TF, adjusting them in opposite directions tends to produce highly similar models. This effect may be contributing to the plethora of local minima. Reducing the number of these may require changing the form of the model, or constraining one of them to a single value. The likely candidate for this is the binding affinity of the TF, which could be directly measured, or modeled (Manke *et al.*, 2008), to assign it a value. By fixing all the TF affinity parameters, we would greatly reduce the complexity of the model, as well as removing all of the local optima corresponding to 'compensating' adjustments of the TF affinity-effectiveness parameter pairs. We intend to explore this idea in future work.

The difficulty of optimizing Reinitz models of transcription does not mean that the (sub-optimally) optimized models provide biological meaningless predictions. As shown in earlier work using the SA algorithm with the LAM cooling schedule, a model optimized using data from *D.melanogaster* can produce accurate predictions when applied to the TFBS-map of other *Drosophila* species (Bauer and Bailey, 2008b). Of practical importance, our faster SA_geom optimization algorithm allows us to train reasonably accurate models of transcription rate in much less time than the LAM cooling version of SA. We have thus achieved our goal of reducing the experimental bottleneck represented by model training, since even sub-optimally trained models can give insight into the relative merits of different models.

ACKNOWLEDGEMENTS

The authors wish to acknowledge John Reinitz for providing the code for the benchmark optimization and Lachlan Dufton for providing the initial implementation of SA.

Funding:Australian Research Council Centre of Excellence in Bioinformatics (to T.L.B.); National Institutes of Health (RO-1 RR021692-01 to T.L.B.); UQ International Research Tuition Award (to D.C.B.).

Conflict of Interest: none declared.

REFERENCES

Bauer,D.C. and Bailey,T.L. (2008a) Stream: Static Thermodynamic REgulAtory Model of transcription. *Bioinformatics*, **24**, 2544–2545.

Bauer,D.C. and Bailey,T.L. (2008b) Studying the functional conservation of cis-regulatory modules and their transcriptional output. *BMC Bioinformatics*, **9**, 220.

Chu,K.-W. *et al.* (1999) Parallel simulated annealing by mixing of states. *J. Comput. Phys.*, **148**, 646–662.

Holland,J.H. (1975) *Adaptation in Natural and Artificial Systems*. University of Michigan Press.

Janssens,H. *et al.* (2006)Quantitative and predictive model of transcriptional control of the Drosophila melanogaster even skipped gene. *Nat. Genet.*, **38**, 1159–1165.

Lam,J.K.-C. (1988) *An Efficient Simulated Annealing Schedule*. PhD Thesis, New Haven, CT.

Manke,T. *et al.* (2008) Statistical modeling of transcription factor binding affinities predicts regulatory interactions. *PLoS Comput. Biol.*, **4**, e1000039.

Prentiss,M.C. *et al.* (2008) Protein structure prediction using basin-hopping. *J. Chem. Phys.*, **128**, 225106.

Reinitz,J. *et al.* (2003) Transcriptional control in Drosophila. *Complexus*, **1**, 54–64.

Riedmiller,M. and Braun,H. (1994) Rprop—description and implementation details. Available at http://citeseer.ist.psu.edu/old/711503.html

Segal,E. *et al.* (2008). Predicting expression patterns from regulatory sequence in Drosophila segmentation. *Nature*, **451**, 535–540.

Stormo,G.D. (1998) Information content and free energy in DNA–protein interactions. *J. Theor. Biol.*, **195**, 135–137.

Zinzen,R.P. and Papatsenko,D. (2007) Enhancer responses to similarly distributed antagonistic gradients in development. *PLoS Comput. Biol.*, **3**, e84.