Hybrid grammar-based approach to nonlinear dynamical system identification from biological time series

B. A. McKinney,[1,3,4] J. E. Crowe, Jr.,[1,2,3] H. U. Voss,[5] P. S. Crooke,[4] N. Barney,[6] and J. H. Moore[6]
[1]*Department of Pediatrics, Vanderbilt University Medical Center, Nashville, Tennessee 37232, USA*
[2]*Microbiology and Immunology, Vanderbilt University Medical Center, Nashville, Tennessee 37232, USA*
[3]*The Program for Vaccine Sciences, Vanderbilt University Medical Center, Nashville, Tennessee 37232, USA*
[4]*Department of Mathematics, Vanderbilt University, Nashville, Tennessee 37232, USA*
[5]*Citigroup Biomedical Imaging Center, Weill Medical College of Cornell University, New York, New York 10021, USA*
[6]*Computational Genetics Laboratory, Department of Genetics, Dartmouth Medical School, Lebanon, New Hampshire 03756, USA*

(Received 13 January 2005; revised manuscript received 17 October 2005; published 22 February 2006)

We introduce a grammar-based hybrid approach to reverse engineering nonlinear ordinary differential equation models from observed time series. This hybrid approach combines a genetic algorithm to search the space of model architectures with a Kalman filter to estimate the model parameters. Domain-specific knowledge is used in a context-free grammar to restrict the search space for the functional form of the target model. We find that the hybrid approach outperforms a pure evolutionary algorithm method, and we observe features in the evolution of the dynamical models that correspond with the emergence of favorable model components. We apply the hybrid method to both artificially generated time series and experimentally observed protein levels from subjects who received the smallpox vaccine. From the observed data, we infer a cytokine protein interaction network for an individual's response to the smallpox vaccine.

DOI: 10.1103/PhysRevE.73.021912 PACS number(s): 82.39.−k, 82.20.−w, 82.20.Wt, 87.16.Yc

I. INTRODUCTION

One of the most important goals of biology is to link genes and their products (mRNA, proteins, and metabolites) into functional pathways and dynamic networks that control intracellular and intercellular processes. Significant progress has been made in achieving this goal through the development of high-throughput technologies in molecular and cellular biology, which have made it possible to measure the gene expression (mRNA level) response to particular stimuli for a large portion of an organism's genome [1,2]. Moreover, it is possible to measure other components of these pathways, such as the level of proteins expressed by a cell or tissue [3] and the concentrations of enzymes, metabolites, and other cell-interaction mediators [4]. Most of this biological data are snapshots of the time-varying biological system. Hence, much of the effort in data analysis has involved the application of statistical and machine learning methods to find genes that discriminate between phenotypes, ignoring the dynamical nature of the biological system. Increasingly, however, experiments are being conducted that sample these systems as a function of time [5–7].

Given the observed time-course data, our goal is to infer a dynamical model of general form

$$\dot{\mathbf{y}}(t) = \mathbf{f}(\mathbf{y}(t), \boldsymbol{\lambda}, \boldsymbol{\epsilon}(t)), \tag{1}$$

where the dimension of the parameter vector $\boldsymbol{\lambda}$ is D_λ, and the length D_y of $\mathbf{y}$ corresponds to the number of different molecular species sampled from the biological system. More explicitly, we are trying to infer a coupled system of D_y first-order nonlinear ordinary differential equations (ODEs) of arbitrary form

$$\begin{aligned}
\dot{y}_1(t) &= f_1(y_1(t), y_2(t), \ldots, y_{D_y}(t); \lambda_1, \ldots, \lambda_{D_\lambda}; \epsilon_1(t))\\
\dot{y}_2(t) &= f_2(y_1(t), y_2(t), \ldots, y_{D_y}(t); \lambda_1, \ldots, \lambda_{D_\lambda}; \epsilon_2(t))\\
&\vdots\\
\dot{y}_{D_y}(t) &= f_{D_y}(y_1(t), y_2(t), \ldots, y_{D_y}(t); \lambda_1, \ldots, \lambda_{D_\lambda}; \epsilon_{D_y}(t)).
\end{aligned} \tag{2}$$

The process noise $\boldsymbol{\epsilon}$ is assumed to be white and normally distributed with covariance matrix Q. We assume that the data vector $\mathbf{z}(t)$ can be written as a deterministic measurement function of the state vector $\mathbf{y}$ plus a Gaussian white-noise term $\boldsymbol{\eta}$ with covariance matrix R

$$\mathbf{z}(t) = \mathbf{G}(\mathbf{y}(t)) + \boldsymbol{\eta}(t). \tag{3}$$

We introduce two grammar-based methods for inferring the parameters $\boldsymbol{\lambda}$ and functional form $\mathbf{f}$ of a nonlinear dynamical model that predicts the behavior of a noisy time series of interacting biomolecules. Grammars allow one to introduce language bias (i.e., a restriction of the search space based on knowledge about the underlying system) by constraining the form of the inferred models. It has been shown that stronger language bias leads to better generalization of learned models [8,9]. Our grammatical method allows one to specify any type of kinetics for the model architecture search: from Michaelis-Menten enzyme kinetics and the S-system approximation down to simple linear approximations. Often one has some intuition into the appropriate model kinetics based on the underlying biochemistry, but if such knowledge is not available, one can simply apply syntactic constraints on the grammar.

Using a genetic algorithm (GA), we test a grammatical evolution approach for nonlinear system identification on a simulated time series. We find that parameter estimation in the nonhybrid approach exhibits slow convergence, hence, we introduce a hybrid (or, more specifically, memetic) approach that combines the model search space capabilities of GAs with an optimal parameter estimation method. A memetic algorithm is an evolutionary algorithm that includes a nongenetic local search or refinement to improve genotypes [10]. The etymologic origin of memetic is the root word "meme," a unit of cultural evolution analogous to a gene from the popular book by Dawkins [11]. The main distinguishing feature of memes is that they are processed and possibly improved by the individual. This approach, however, is still Darwinian because, unlike Lamarckian evolution, the acquired trait is not directly inherited. Rather, what is inherited in our case are model architectures with the ability to improve over the lifetime of the model. In our memetic algorithm, we implement the unscented Kalman filter (UKF) [12], which has been demonstrated to be an accurate and computationally efficient method for parameter estimation of nonlinear models from noisy time series [13]. However, the UKF signal analysis method does not estimate the functional form of the model. Thus, the grammatical memetic evolution approach introduced in this paper is an ideal hybridization of methods for noisy nonlinear system identification.

We apply the hybrid technique to infer a phenomenological dynamic model from a sparse time series of protein expression from the directed analysis of cytokines measured from the serum of subjects who received the smallpox vaccine. Cytokines are small protein molecules that are central to communication among immune system cells and between immune cells and other tissue cell types. Cytokines act by binding to their cell-specific receptor. These receptors are located in the cell membrane and initiate a signal cascade that eventually leads to biochemical and phenotypic changes in the target cell. Before an immune cell can kill a foreign antigen, such as a virus, bacterium, pollen, or tumor, the cell must first be recruited and activated. Recruitment is frequently mediated by chemotactic cytokines (i.e., chemokines), while activation is often induced by cytokines such as interferon (IFN), tumor necrosis factor (TNF), and interleukins (ILs), which are produced by a variety of cell types including natural killer (NK) cells and T lymphocytes. In response to virus infection, cytokines may induce cell division, differentiation, programmed cell death (apoptosis), activation, or movement, and can even upregulate other cytokines, thus, constituting a network of interacting cytokine proteins.

The outline of this paper is as follows. We begin by introducing grammatical evolution for nonlinear dynamic system identification. Then we review the unscented Kalman filter and combine it with grammatical evolution to create a grammar-based memetic algorithm. We compare these two methods by using each to identify the model architecture and parameters for a simulated time series involving a nonlinear feedback loop, and we find the memetic approach outperforms the nonhybrid approach. Finally, we apply the grammatical memetic evolution method to observed time-series cytokine protein levels from subjects who received the Aventis Pasteur smallpox vaccine.

II. GRAMMATICAL EVOLUTION

Grammatical evolution (GE) is an evolutionary algorithm (EA) in which a Backus-Naur form (BNF) grammar is specified that allows a program or model to be constructed from a genetic algorithm (GA) bit string [14]. GAs are a robust set of search techniques that perform well on a wide class of problems. Specifically, GAs have been shown to be a robust approach to generating network structures and models [15,16]. A grammar is a set of production rules that can be chosen from the GA bit string to produce sentences in any language. Sentences created by our grammar are systems of coupled nonlinear differential equations. BNF is a formal notation for describing the syntax of a context-free grammar as a set of production rules, consisting of terminals (model elements) and nonterminals (the production rules themselves) [17]. We now discuss the mapping from GA bit strings to GE nonlinear models. For a more detailed introduction to grammars in the context of genetic algorithms, we refer the reader to Ref. [14].

In our discussion of grammars, we use the convention in which nonterminals are enclosed in angle brackets (e.g., ⟨nonterminal⟩). Equation (4) illustrates the type of grammar production rules we will use to infer a nonlinear protein interaction model from immunologic time-series data

$$\begin{aligned}\langle\text{model-expr}\rangle ::= {} & \langle\text{param}\rangle\langle\text{var}\rangle + \langle\text{param}\rangle\langle\text{var}\rangle \qquad (0)\\ & \mid \ \langle\text{param}\rangle\langle\text{reg-fn}\rangle + \langle\text{param}\rangle\langle\text{var}\rangle \quad (1).\end{aligned} \tag{4}$$

When there is more than one choice for a rule, the choices are delimited by a vertical bar with the number of the choice given in parentheses to the right. For each dependent variable y_i (denoted generically by ⟨var⟩), a nonlinear differential equation expression [i.e., $\mathbf{f}$ in Eq. (1)] is constructed from ⟨model-expr⟩ using the GA bit string. This is done by breaking up the GA bit string into 16-bit segments and converting the bit string into a sequence of 16-bit integers, referred to now as "codons," while the wrapping property of modular arithmetic is used to decode each nonterminal from the GA codon. The choice of rule used for each nonterminal is given by rule=(codon mod M), where M is the number of possible rules for the nonterminal being considered. After one of the two rules for ⟨model-expr⟩ is chosen, its nonterminals are recursively replaced with the corresponding rules, based on the remaining GA codons, until ⟨model-expr⟩ is given only in terms of terminals. The final GE individual is a system of coupled nonlinear differential equations, whose fitness is determined by measuring the nearness in a least-squares sense of the numerical prediction and the observed expression levels at the available time points. One can allow each node in the system to have a different number of connections by including recursive elements in the grammar.

The connectivity of the model is determined by the ⟨var⟩ nonterminals. Whenever ⟨var⟩ is encountered in the grammar, its value is given by the terminal y_i, where i =(codon mod N), N being the number of observed biomolecules in the data set. In Eq. (4), the GA is given two choices for ⟨model-expr⟩: a purely linear equation (0) or an equation containing a nonlinear regulatory term (1). The presence of ⟨var⟩ in ⟨reg-fn⟩ of Eq. (5) allows the GA codons to determine the influence of a protein (e.g., as a lymphocyte inhibitor in a cytokine network or a transcription factor in a gene regulatory network) on other elements of the network. Biomolecule y_i, an instantiation of ⟨var⟩, can activate or inhibit the target biomolecule, depending on whether the GA codon yields rule (0) or (1) for the regulation function ⟨reg-fn⟩ in Eq. (5)

$$\begin{aligned}\langle\text{reg-fn}\rangle ::= {} & h^{+}(\langle\text{var}\rangle,\langle\text{param}\rangle) \quad (0)\\ | {} & h^{-}(\langle\text{var}\rangle,\langle\text{param}\rangle) \quad (1).\end{aligned} \tag{5}$$

The Hill functions $h^{+}(y,\theta)=y/(y+\theta)$ and $h^{-}(y,\theta)=1-h^{+}(y,\theta)$ model the effect of biomolecule y activating or inhibiting, respectively, the target biomolecule.

In the GE (nonhybrid) method, whenever a ⟨param⟩ nonterminal is encountered, its value in the interval $(R_{\min},R_{\max}]$ is given by $(R_{\max}-R_{\min})/\text{codon}+R_{\min}$. To our knowledge, this mapping from codons into an interval is a new approach in grammatical evolution constant creation. Other constant creation grammars, such as digit concatenation [18], have been introduced, but we use a direct codon mapping method because it evolves constants directly from a single codon. One drawback of specifying an interval is that one may not *a priori* know its range; however, this limitation can be overcome by including a recursive element in the constant creation grammar to allow real constants to evolve outside the original interval. The ability to evolve real constants more easily is an advantage of GE over genetic programming, which directly evolves programs but has difficulties with constant creation [19].

III. GRAMMATICAL MEMETIC EVOLUTION

With a biologically motivated grammar specified, the GA is in charge of evolving the model structure and connections. In the GE approach, evolution provides a balanced search mechanism for the model architecture and parameter spaces to arrive at an approximate model. However, as we show in Sec. IV, this method is slow to converge to the optimal parameter values with precision. To speed up parameter estimation, we combine the grammar-based GA with the unscented Kalman filter (UKF) [12]—an optimal recursive parameter estimation method—to create a more powerful method we call grammatical memetic evolution (GME). One reason for introducing the nonhybrid approach in the previous section is for comparison purposes to show that premature convergence is not an issue for our memetic approach.

For the GME approach, whenever a ⟨param⟩ nonterminal is encountered, an unknown parameter is inserted into the model expression and the parameters are later optimized by the UKF, which is the nongenetic refinement algorithm we use in our memetic algorithm. The UKF is itself a hybrid approach that unites the accuracy of Monte Carlo Markov chain particle filters with the speed of traditional Kalman filters. This unification is achieved through "deterministic sampling" [12,13]. Another advantage of the UKF is that there is no need to calculate derivatives with respect to the state variables, which would be challenging in this inference approach wherein the model architecture is not fixed but allowed to evolve. The UKF is robust to measurement noise, a perennial challenge of biological data, and the UKF can naturally deal with unobserved variables, a particular challenge for modeling biological networks, such as gene regulatory networks, in which gene and protein expression data often are not measured in a coordinated fashion.

The following are the key elements of Kalman filtering for parameter estimation. Because the observed time series is sampled at discrete time points, we rewrite Eq. (1) as a discrete nonlinear deterministic state space model [20] for the state at observed time point t_{k+1} in terms of its predecessor at time t_k

$$\mathbf{y}_{k+1}=\mathbf{F}(\mathbf{y}_k,\boldsymbol{\lambda}_k,\boldsymbol{\epsilon}_k), \tag{6}$$

and

$$\mathbf{F}(\mathbf{y}_k,\boldsymbol{\lambda}_k,\boldsymbol{\epsilon}_k)=\mathbf{y}_k+\int_{t_k}^{t_{k+1}}\mathbf{f}(\mathbf{y}(T),\boldsymbol{\lambda},\boldsymbol{\epsilon}(T))dT, \tag{7}$$

where the states obey the Markov condition that each state follows uniquely from its predecessor. For parameter estimation, it is convenient to represent the state of the system augmented by the vector of parameters $\boldsymbol{\lambda}_k$

$$\mathbf{x}_k=\begin{pmatrix}\boldsymbol{\lambda}_k\\ \mathbf{y}_k\end{pmatrix}. \tag{8}$$

The recursive engine of the Kalman filter [21] involves correcting the predicted moments with the observed data using equations

$$\hat{\mathbf{x}}_{k+1|k+1}=\tilde{\mathbf{x}}_{k+1|k}+\mathbf{K}_{k+1}(\mathbf{z}_k-\tilde{\mathbf{y}}_{k+1|k}) \tag{9}$$

and

$$\mathbf{K}_{k+1}=\mathbf{P}_{\tilde{\mathbf{x}}\tilde{\mathbf{y}}}\mathbf{P}_{\tilde{\mathbf{y}}\tilde{\mathbf{y}}}^{-1}. \tag{10}$$

The *a posteriori* estimate of the augmented state at time step $k+1$, given by Eq. (9), consists of the *a priori* prediction $\tilde{\mathbf{x}}_{k+1|k}$ at the previous time step and a correction term proportional to the difference between the observed data $\mathbf{z}_k$ and the estimate of the unaugmented state $\tilde{\mathbf{y}}_{k+1|k}$ at the previous step. The Kalman gain or blend matrix $\mathbf{K}$, updated by Eq. (10), is chosen to minimize the trace of the *a posteriori* error covariance matrix $\mathbf{P}_{\hat{\mathbf{x}}\hat{\mathbf{x}}}$ because the trace of this covariance matrix equals the sum of the squared errors of the components of the posterior estimate of $\mathbf{x}$ (i.e., $\hat{\mathbf{x}}_{k+1|k+1}$). In Eq. (10), $\mathbf{P}_{\tilde{\mathbf{x}}\tilde{\mathbf{y}}}$ is the covariance matrix for the deviation of the $\mathbf{x}$ and $\mathbf{y}$ states from their *a priori* estimates. Together with the unscented transformation below, equations (9) and (10) are used recursively to improve the state and parameter estimates by stepping through the experimentally observed time points t_k until the final time point is reached. We perform multiple sweeps through the time-ordered data until the parameter estimates converge between sweeps to within a specified tolerance.

The unscented transformation retains the exact nonlinearity of the model $\mathbf{F}$ but approximates the *a posteriori* density of the state $\mathbf{x}_{k+1}$ by a Gaussian. Instead of linearizing using

TABLE I. Inferred model parameters. Target: parameters used in the nonlinear Eq. (14) model to simulate sparse time-series data. Both grammatical evolution (GE) and grammatical memetic evolution (GME) found the correct network topology, but GME found the correct topology more efficiently and found the network parameters more precisely than GE.

	$\kappa_{1,3}$	$\kappa_{2,1}$	$\kappa_{3,2}$	$\theta_{1,3}$	γ_1	γ_2	γ_3
Target	0.9	1.0	0.6	0.9	1.0	0.6	0.8
GE	0.923	0.984	0.639	0.978	0.927	0.584	0.861
GME	0.904	0.997	0.600	0.907	1.003	0.598	0.800

Jacobians, the UKF algorithm uses a set of $2D_x+1$ $(D_x=D_y+D_\lambda)$ sample points (called σ points) to parametrize the means and covariances, and then one propagates the σ points through the state equations. Consider a normally distributed random variable $\mathbf{r}$. Such a random variable is completely described by its mean $\bar{\mathbf{r}}$ and covariance matrix $\mathbf{P}$. This information can be stored with some redundancy in $2D_r$ sigma point matrices $\boldsymbol{\chi}_i$ whose columns are computed by

$$\boldsymbol{\chi}_0 = \bar{\mathbf{r}}, \tag{11}$$

$$\boldsymbol{\chi}_i = \bar{\mathbf{r}} + (\sqrt{D_r}\mathbf{P})_i, \quad (i = 1, \ldots, D_r), \tag{12}$$

$$\boldsymbol{\chi}_j = \bar{\mathbf{r}} + (\sqrt{D_r\mathbf{P}})_j, \quad (j = D_r + 1, \ldots, 2D_r), \tag{13}$$

where we use the Cholesky decomposition to find the matrix square root, though any choice is suitable, and $(\sqrt{\cdot})_i$ denotes the ith row or column of the matrix square root. Theoretical details and application of the unscented transformation algorithm to Kalman filtering for state space modeling and the deterministic calculation of the statistics of a random variable undergoing a nonlinear transformation can be found in Refs. [12,13,22].

IV. NONLINEAR SYSTEM IDENTIFICATION FOR A MODEL SYSTEM

As a proof-of-principle test, we use the grammar-based methods (GE and GME) to evolve a nonlinear model from data simulated for an $N=3$ one-gene inhibitory feedback loop based on the operon model

$$\frac{dy_1}{dt} = \kappa_{1,3}h^{-}(y_3, \theta_{1,3}) - \gamma_1 y_1,$$

$$\frac{dy_2}{dt} = \kappa_{2,1}y_1 - \gamma_2 y_2,$$

$$\frac{dy_3}{dt} = \kappa_{3,2}y_2 - \gamma_3 y_3. \tag{14}$$

We corrupt the simulated data with external Gaussian observation noise $\boldsymbol{\eta}$ with constant covariance $R=0.01$, but we assume no intrinsic noise in the model, $\boldsymbol{\epsilon}=0$. We simulate six time points for each quantity, which is a typical sampling frequency for sparse biological time-series data sets. The "Target" row of Table I shows the model parameters used to simulate the data plotted in Fig. 1. First introduced in Ref. [23] and then extended in Ref. [24], the operon model continues to be a useful framework for modeling biological sys-

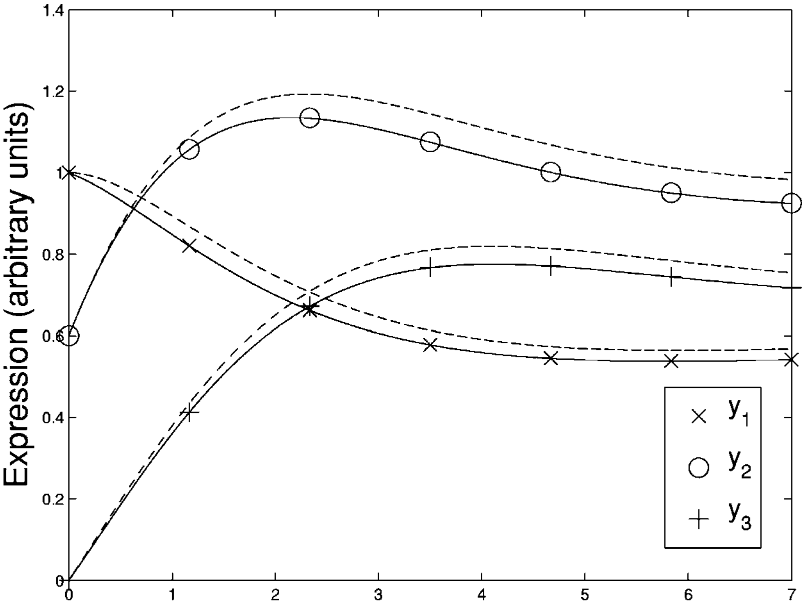

FIG. 1. Simulated gene product expression versus time in days for a one-gene feedback loop. Units vary in experimental expression data, but a natural unit is mRNA or protein copies per cell. Symbols indicate time points simulated from Eq. (14) with target parameters from Table I. Dashed lines are the numerical solutions of the ODE system evolved by grammatical evolution (GE) at generation 1000 and solid lines by grammatical memetic evolution (GME) at generation 100.

tems [25,26]. In Eq. (14), the structural gene that codes for a protein or enzyme is linked with an operator gene y_1 that regulates transcription and represents the expression of mRNA. The constants $\kappa_{i,j}$ are production constants and the parameters γ_i are degradation constants for the operator gene and the other products of the structural gene. The expression of molecule y_i $(i>1)$ increases in proportion to the previous molecule y_{i-1} in the loop and decreases in proportion to its own expression through degradation, diffusion, and growth dilution. To close the feedback loop, the operator gene y_1 is regulated by the effector molecule y_N $(N=3)$ via the inhibitory Hill function h^{-}.

Methods were implemented in C^{++} using the genetic algorithm library GALIB [27], which we modified for parallel use on a LINUX cluster with the message-passing interface [28]. We use a crossover rate of 0.8 and mutation rate of 0.2. Because there are many systems that are not well behaved in the differential equation model search space, we initialize the populations so that all individuals at the initial generation are valid, and we use a steady-state GA mechanism to make invalid individuals less likely to be passed on to later generations [29]. In the steady-state GA, 10% of the least fit individuals from each population are replaced by offspring resulting from crossover and mutation of the fittest individuals, unlike a generational GA where the entire population is replaced each generation. For the nonhybrid GE, we ran the GA for 1000 generations with 10 populations of 400 individuals (ODE systems) with migration of the most fit solution from each population to all other populations every 10 generations. For the GME approach, only one population of 100 individuals was used, thus, no migration was needed. Due to the sparsity of typical biological time series, the UKF algorithm requires several iterations through the data to achieve convergence of model parameters. Optimal parameters can often be obtained in one sweep through time series with higher sampling frequency.

Within the operon-model grammar, GE (nonhybrid) was able to traverse the search space and identify the correct connectivity and functional form of the model, the correct variables responsible for gene regulation, and the approximate strengths of production and degradation of gene products. However, the infinite parameter search space leads to slow convergence of the model parameters (see Sec. IV A for more details on the dynamics of the GE model evolution). The GME approach exhibits a dramatic improvement over the nonhybrid approach. Not only were the GME-inferred model parameters found with much higher precision (see Table I), but they were also found with much less computing power. The GME method found the correct model in less than 100 generations using one population (single processor) with only 100 individuals, which represents a large reduction in the number of fitness function evaluations. This is also evidence that the GME method does not prematurely converge to a local optimum for this nonlinear system identification problem.

A. Dynamics of evolution

Figure 2 shows the fitness of the best individual at each generation for a typical nonhybrid GE run. The stair-step appearance is due in part to the delay in the migration frequency. This periodic migration allows each population to build upon the evolutionary progress made by other populations by recombination of useful solution traits. Migration also enables populations to more easily escape a local optimum. In this particular run, the functional form of the model was found by generation 100. The inset of Fig. 2 shows more detail from this run for the time interval 0 to 100 generations. The jumps in fitness seen in the inset of Fig. 2 are due to the emergence of correct network connections. After the correct model architecture is found, there is a period of slow but steady increase in fitness as these favorable structures are disseminated throughout all populations. In the terminology of Ref. [30], this period would be called the first "epoch of innovation." During this epoch, the dissemination of the favorable network connections allows for later rapid increases in fitness due to improvements in the rate constants. For example in Fig. 1, the doubling of the fitness during generation 360 to 400 is due to the considerable improvement of two model parameters while preserving the correct model connectivity found at generation 100. In contrast, the memetic algorithm is able to find a very precise solution in less than 100 generations using a much smaller population size.

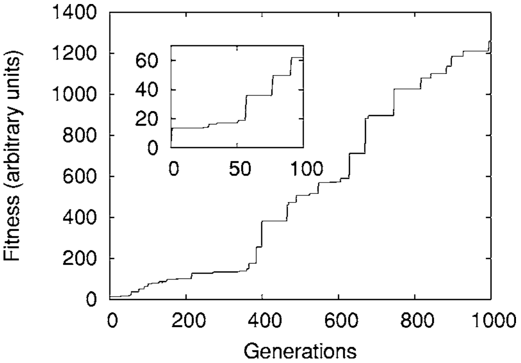

FIG. 2. Fitness of the best-of-generation individual among all populations versus the number of generations using the GE method. A rapid increase in fitness is observed around generation 400 as favorable model elements have had time to filter through the populations. Inset: Detailed look at evolution through generation 100 at which time the correct functional form of the model is discovered.

V. APPLICATION TO IMMUNOLOGIC DATA

We now apply GME to observed time-series cytokine protein concentration data (shown in Fig. 3) from subjects who received the Aventis Pasteur smallpox vaccine. The model was inferred by the GME method from human serum cytokine levels measured after smallpox immunization. Cytokine levels were measured at seven time points over a month, resulting in the directed analysis of four cytokines. These systemic cytokines, representing functional subsets T_H1 (TNF-α and IL-2) and T_H2 (IL-4 and IL-10), were measured using a sensitive flow cytometric bead array analysis that allowed for multiple cytokine analyses from a single sample [31]. The study was reviewed and approved by the Vander-

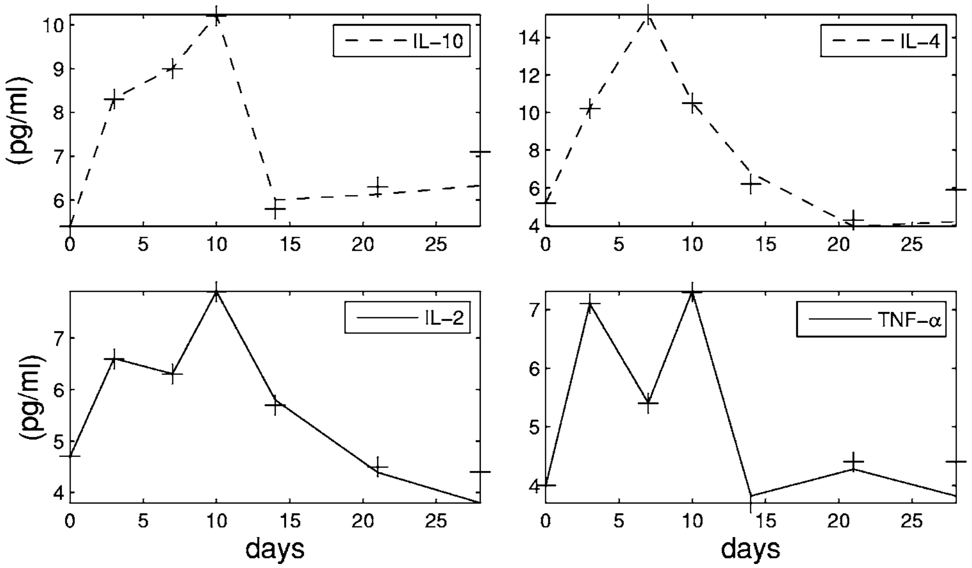

FIG. 3. Serum cytokine concentrations in (pg/ml) following smallpox immunization and interpolations between the posterior model predictions at the observed time points. Dashed lines: state estimates for T_H2 cytokines, IL-10 and IL-4. Solid lines: state estimates for T_H1 cytokines, IL-2 and TNF-α.

bilt Institutional Review Board, and samples were obtained from volunteers following informed consent. The solid and dashed lines in Fig. 3 are interpolations between the *a posteriori* Kalman filter predictions of the state at the observed time points for the final run of the GME algorithm with a linear grammar. The predictor-corrector nature of the Kalman filter causes the estimated cytokine trajectories to obey constraints at each sampled time step.

In a typical experiment, either protein or cell concentrations are available, but they are usually not measured in a coordinated fashion. However, it has been shown that linear ODEs are capable of discovering phenomenological interaction networks when elements from the full network are unobserved [32]. Furthermore, when using a biologically realistic nonlinear grammar, such as the operon grammar, network connections between biochemicals are often parametrized by more than one number (e.g, Hill parameters), making it more difficult to interpret a quantitative network diagram. Thus, to obtain the network diagram in Fig. 4, we used a linear ODE grammar. GME predicts causal connections between the cytokines and allows for the existence of loops in the network topology when warranted by the data. Bayesian networks do not allow loops as they prevent the joint probability distribution of the estimated network from being decomposed into a product of conditional probability distributions [33]. Moreover, GME provides a quantitative model of the dynamic system. In the network model of Fig. 4, T_H1 cytokines are in the bottom row of nodes and T_H2 along the top row. The T_H1 response to an antigen produces a cytokine profile that supports inflammation and primarily activates certain T cells and macrophages, while the T_H2 response mainly activates B cells and an antibody-dependent immune response. It is interesting to note that our preliminary model predicts that the cytokines within each T_H type do not influence each other directly. The feedback loop between IL-2 and IL-4 suggests these cytokines as a bridge across T_H types. On a related note, our algorithm identifies the well established fact that IL-4 is inhibitory to T_H1 responses.

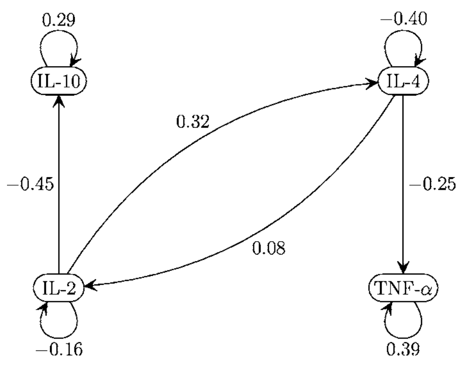

FIG. 4. Cytokine protein interaction network inferred from human serum cytokine levels following smallpox vaccination. Bottom row of nodes: T_H1 cytokines. Top row of nodes: T_H2 cytokines.

VI. DISCUSSION

We introduced a flexible, grammar-based method that searches the space of model components and connectivities and estimates the parameters of the nonlinear differential equation models inferred from sparse biochemical time-series data. Grammatical memetic evolution (GME) was able to discover the correct form of the model and precise, unbiased model parameters from simulated data with very sparse sampling. By comparing GME with a nonhybrid method (GE), our simulation studies also provide evidence that the memetic algorithm is not prone to premature convergence for nonlinear dynamical system identification. We applied this method to immunologic time-series data to identify a dynamical model for an individual's cytokine response to smallpox vaccine.

The grammar can be tailored to the biochemical system under analysis if pathway or mechanistic information is known about the system. New grammar rules, such as products of variables or more general operations, can be created simply by modifying a text file, as opposed to recompiling source code. If there is little prior knowledge of the system, or if one wants GME to explore exotic biochemical mechanisms, one can simply apply syntactic constraints on the grammar definition. A future direction is to incorporate prior biological knowledge from databases, such as KEGG and Biocarta, into the fitness function. This type of strategy has been used in network inference algorithms [34] and may reduce the effect of noise and improve the power to predict network connections.

The NP completeness of identifying an interaction network with high accuracy makes it an intrinsically difficult problem. Due to this difficulty, we implemented heuristic and meta-heuristic search methods. While the current paper focused on the directed analysis of a small subset of biomolecules, another future direction for larger, array-based experiments is to use clustering and filtering strategies to reduce the dimensionality of these high-throughput data sets [35]. We are also developing a hybrid of UKF with genetic programming [36], another effective evolutionary algorithm for equation discovery.

In this paper, our simulations were based on deterministic models with additive external measurement noise. However, there has been evidence to suggest that in some biological systems, intrinsic noise may play an important role in determining cell phenotypes [37]. Some fluctuations observed in the cytokine kinetics may be due to intrinsic stochastic mechanism. The GME method is able to accommodate intrinsic noise in the underlying model, and, thus, a future direction will be to determine the role of intrinsic noise strength on the inference of stochastic models.

ACKNOWLEDGMENTS

This work was supported by NIH Grants No. K25 AI-64625, AI-59694, and AI-57661. Generous support was also provided by the Vanderbilt Program in Biomathematics.

[1] M. Schena, D. Shalon, R. Davis, and P. Brown, Science **270**, 467 (1995).

[2] D. Lockhart, H. Dong, M. Byrne, M. T. Follettie, M. V. Gallo, M. S. Chee, M. Mittmann, C. Wang, M. Kobayashi, H. Norton, and E. L. Brown, Nat. Biotechnol. **14**, 1675 (1996).

[3] S. P. Gygi and R. Aebersold, Curr. Opin. Chem. Biol. **4**, 489 (2000).

[4] L. Raamsdonk *et al.*, Nat. Biotechnol. **19**, 45 (2001).

[5] R. Cho *et al.*, Mol. Cell **2**, 65 (1998).

[6] X. Wen, S. Fuhrman, G. Michaels, D. Carr, S. Smith, J. Barker, and R. Somogyi, Proc. Natl. Acad. Sci. U.S.A. **95**, 334 (1998).

[7] S. E. Baranznin *et al.*, PLoS Biol. **3**, 166 (2005).

[8] C. Nedellec, C. Rouveirol, H. Ade, F. Bergadano, and B. Tausend, in *Advances in Inductive Logic Programming*, edited by L. D. Raedt (IOS Press, Amsterdam, 1996), Vol. 32, pp. 82–103.

[9] V. Vapnik and Y. Chervonekis, Theor. Probab. Appl. **26**, 532 (1981).

[10] P. Moscato, *Memetic Algorithms: A Short Introduction* (McGraw-Hill, London, 1999), pp. 219–234.

[11] R. Dawkins, *The Selfish Gene* (Oxford University Press, New York, 1976).

[12] S. Julier, J. Uhlmann, and H. F. Durrant-Whyte, IEEE Trans. Autom. Control **45**, 477 (2000).

[13] A. Sitz, U. Schwarz, J. Kurths, and H. U. Voss, Phys. Rev. E **66**, 016210 (2002).

[14] M. O'Neill and C. Ryan, IEEE Trans. Evol. Comput. **5**, 349 (2001).

[15] X. Yao, Int. J. Intell. Syst. **8**, 539 (1993).

[16] C. W. Chau, S. Kwong, C. K. Diu, and W. R. Fahrner, in IEEE International Conference on Acoustics, Speech, and Signal Processing, Vol. 3 (IEEE, New Jersey, 1997), pp. 1727–1730.

[17] M. Marcotty and H. Ledgard, *The World of Programming Languages* (Springer-Verlag, New York, 1986).

[18] I. Dempsey, M. O'Neill, and A. Brabazon, in GECCO'98: Proceedings of the Genetic and Evolutionary Computation Conference, edited by W. Banzhaf *et al.*, (Morgan Kaufman, San Francisco, 2004), pp. 447–458.

[19] M. Evett and T. Fernandez, in Genetic Programming 1998: Proceedings of the Third Annual Conference, edited by J. R. Koza, (Morgan Kaufman, San Francisco, 1998), pp. 66.

[20] A. Gelb, *Applied Optimal Estimation* (The MIT Press, Cambridge, MA, 1974).

[21] R. E. Kalman, J. Basic Eng. **82**, Series D, 35 (1960).

[22] H. U. Voss, J. Timmer, and J. Kurths, Int. J. Bifurcation Chaos Appl. Sci. Eng. **14**, 1905 (2004).

[23] F. Jacob and J. Monod, Cold Spring Harbor Symp. Quant. Biol. **26**, 193 (1961).

[24] J. Tyson and H. Othmer, Prog. Theor. Biol. **5**, 2 (1978).

[25] P. Smolen, D. A. Baxter, and J. Byrne, Bull. Math. Biol. **62**, 247 (2000).

[26] H. de Jong, J. Comput. Biol. **9**, 67 (2002).

[27] M. Wall, GALIB, http://lancet.mit.edu/ga/ (1995).

[28] W. Gropp, E. Lush, and A. Skjellum, *Using MPI: Portable Parallel Programming with the Message-Passing Interface*, 2nd ed. (MIT Press, Cambridge, MA, 1999).

[29] M. O'Neill and C. Ryan, in *Grammatical Evolution: Evolutionary Automatic Programming in an Arbitrary Language (Genetic Programming Series) Vol. 4*, (Kluwer Academic, 2003).

[30] M. Mitchell, J. Crutchfield, and P. Hraber, Physica D **75**, 361 (1994).

[31] M. T. Rock, S. M. Toder, T. F. Talbot, K. M. Edwards, and J. E. C. Jr., J. Infect. Dis. **189**, 1401 (2004).

[32] J. Tegnér, Proc. Natl. Acad. Sci. U.S.A. **100**, 5944 (2003).

[33] M. de Hoon, S. Imoto, K. Kobayashi, N. Ogasawara, and S. Miyano, in Proceedings of the Pacific Symposium on Biocomputing, edited by R. B. Altman *et al.*, (World Scientific, New Jersey, 2003), pp. 17.

[34] A. Shin and H. Iba, Genome Informatics **14**, 94 (2003).

[35] M. Wahde and J. Hertz, BioSystems **55**, 129 (2000).

[36] J. Koza, *Genetic Programming: On the Programming of Computers by Means of Natural Selection* (MIT Press, Cambridge, MA, 1992).

[37] M. Koern, T. C. Elston, W. J. Blake, and J. J. Collins, Nat. Rev. Genet. **6**, 451 (2005).