Learning Algorithm for Modeling Complex Spatial Dynamics

Thomas P. Meyer, Fred C. Richards, and Norman H. Packard
Center for Complex Systems Research and the Physics Department, Beckman Institute, University of Illinois, 405 North Mathews Avenue, Urbana, Illinois 61801 [a]
(Received 18 October 1988)

A learning algorithm is developed to build a dynamical model from complex spatial data consisting of discrete values on a lattice evolving in time. The resulting dynamical system is a cellular automation, and may be used for forecasting or for regenerating global spatial patterns. Part of the learning algorithm is a novel application of the genetic algorithm, originally developed in the field of machine learning. We outline extensions of this method to construct models for spatial dynamics that have continuous variables as well.

PACS numbers: 89.70.+c, 02.70.+d, 06.50.Dc

In this paper, we consider the experimental analysis of complex spatial patterns. Efforts in this direction have only begun;[1-6] our work with spatial data is based on the use of a learning algorithm to build a model from data. This approach has been implemented for time-series analysis,[1,7-10] and suggested for the analysis of spatial data.[1]

We present a learning algorithm designed to build models from spatial data that consist of a sequence of two-dimensional images with discrete values at discrete lattice sites. The spatial extent of the data makes the problem more complicated than for the time-series data, but the discreteness simplifies the problem. After describing a learning algorithm for this type of data and testing it on artificially generated data, we will comment on combining this approach with that developed for continuous-variable data. The resulting learning algorithms will search an enlarged space of models, and will enable us to further close the gap between experiment and theory.

The experimental system at which this analysis is aimed is dendritic solidification (this application will appear in a later paper). In this case, binary values at each lattice site are obtained by denoting the presence of solid by one and the absence of solid by zero. In other cases, binary data may be obtained by data-reduction techniques.[11] In any case, we will assume that the data take the form of a temporal sequence of patterns of ones and zeros on a lattice. We consider the two-dimensional case because we aim to use the techniques for two-dimensional experimental data. The analogous formulation for one dimension is completely straightforward; for three dimensions the generalization is also straightforward, but the data-gathering technology typically does not exist.

As in the case of the time-series data mentioned above, the object of the learning algorithm will be to construct a dynamical model for the observed data. This is accomplished by searching a space of dynamical rules. We will assume spatial locality for the dynamical rule that takes patterns to patterns. A hypothetical deterministic rule would have the form of a cellular-automaton (CA) rule.[12] The possibility of "inertial effects" in the dynamics suggests the consideration of rules for which the future depends on more than a single past state. When the inevitable presence of noise is included, we find that the task of the learning algorithm is to search through the space of probabilistic CA rules. Such a rule is given by the conditional probability distribution $P(s \mid s_1, \ldots, s_n)$, with s being the future value of the site that we wish to predict and $s_1, \ldots, s_n$ being the values of the n sites of a space-time neighborhood template in the past. Thus, in our context, there is a one-to-one correspondence between rules and templates used to construct $P(s \mid s_1, \ldots, s_n)$.

Though we would like to search through arbitrary templates in space-time, computational limitations lead us to choose a template as large as memory allows, and then to explore the space of subtemplates. We chose the two-time-step master template illustrated in Fig. 1. Given a large temporal sequence of images, $P(s \mid s_1, \ldots, s_n)$ is approximated by collecting a frequency-of-occurrence histogram.

The conditional probability histogram $P(s \mid s_1, \ldots, s_n)$

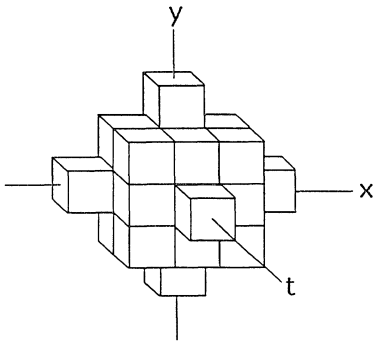

FIG. 1. The template in space-time that is searched by the learning algorithm. Mutual information was computed between the future cell and the nearby cells for two time steps in the past.

is itself a probabilistic dynamical rule, and may be iterated and compared to the results of the actual data. It is not, however, necessarily the simplest rule that could be constructed from the data; to see this, consider constructing $P(s \mid s_1, \ldots, s_n)$ for test data generated by the identity rule (simply a fixed pattern repeated in time). The future value depends only on a single cell of the past template, so all the remaining entries in the distribution are irrelevant. The learning algorithm should seek this out and find the simplest template that still captures the essential dynamics.

A measure of the relevance of past-site values to the future-site value is given by the mutual information.[13-16] It is computed for a subtemplate $(s_1, \ldots, s_m)$ [where these are a subset of the original $(s_1, \ldots, s_n)$ with $m < n$] from $P(s \mid s_1, \ldots, s_m)$ by

$$I(s;s_1, \ldots, s_m) = \sum P(s,s_1, \ldots, s_m) \log_2 \frac{P(s,s_1, \ldots, s_m)}{P(s)P(s_1, \ldots, s_m)},$$

where the joint distribution $P(s,s_1, \ldots, s_m)$ is related to the conditional distribution $P(s \mid s_1, \ldots, s_m)$ by

$$P(s,s_1, \ldots, s_m) = P(s \mid s_1, \ldots, s_m) P(s_1, \ldots, s_m);$$

the sum is taken over all configurations $(s_1, \ldots, s_m)$ and both possible values of s. Since we are searching for a rule which describes the behavior of a single site of the lattice, the maximum amount of mutual information that can be extracted for neighboring space-time templates about a single site will be the amount of information measured from observing the behavior of single sites. The maximum amount of mutual information a space-time template can provide is $I_{\max} = H(s)$, where $H(s)$ is defined as $H(s) \equiv -\sum P(s) \log_2 P(s)$, the sum taken over all possible values of s (0 or 1). The learning algorithm should seek a rule given by a template that yields mutual information as close to $I_{\max}$ as possible.

A brute force approach would be to try all possible templates to find the optimal one. However, there are so many that this is infeasible, so the space of templates must be searched by some other means. Another approach is to build up a template cell by cell by finding the single cell that has the highest mutual information with the future, fixing it, and looking for a second cell which may be added to the first to give the highest mutual information, and so on. This approach is not satisfactory because it cannot detect "many-body" informational correlations that are present in general.

The *genetic algorithm*,[17] a technique developed in machine learning, is well suited for the task of searching the space of templates, given that each template has an associated "fitness." In this context, application of the genetic algorithm begins with a set of test templates chosen at random. The learning takes place by an evolutionary dynamic: Every generation the templates are ordered by fitness and the less fit ones discarded. The poor templates are then replaced by new ones obtained from the old ones by the application of two genetic operators: point mutation, which corresponds to adding a new cell to a template or taking one away; and crossover, which produces two new templates from two old ones by making an arbitrary cut in space-time that divides the templates into two pieces, and cross matching the pieces. Repeating the evaluation, ranking, and replacement evolves the set of templates to be increasingly fit. We can think of the fitness function as defining a surface, or "landscape," and the genetic operators move us about on the landscape. Point mutation therefore corresponds to a local displacement on the landscape, while crossover produces the larger jumps needed to seek out a global maximum of the landscape.

Clearly, it is crucial that we choose the fitness function judiciously if the genetic algorithm is to be an effective tool. As we stated above, our algorithm should find a subtemplate which maximizes $I(s;s_1, \ldots, s_m)$. On the other hand, as the number of sites in a subtemplate, m, increases, $I(s;s_1, \ldots, s_m)$ becomes increasingly difficult to evaluate for finite data sets. For example, if we were to record simultaneously the outcomes of two random events, we would expect the mutual information between these two (finite) sets of measurements to be zero. It can be shown, however, that if event 1 has X possible outcomes and event 2 has Y possible outcomes, and we make only N simultaneous measurements, the mutual information that we estimate using a frequency-of-occurrence histogram will not be zero but will be proportional to $(X-1)(Y-1)/N$. In our application $X=2$, and if we consider a subtemplate comprising m sites, $Y=2^m$. Our fitness function should weigh any increase in $I(s;s_1, \ldots, s_m)$ that we gain by increasing our template size, m, against the increased inaccuracy of our statistics.[18] We have therefore defined our fitness function as $F = I - 2^m/N$. In the ideal situation where $N \to \infty$, we have $F = I$ and the genetic algorithm should search out a template that maximizes $I(s;s_1, \ldots, s_m)$. In this limit, our fitness function is the same, up to an added constant, as the model criterion used in Ref. 1, based on indeterminacy. For finite-size data sets, however, the fitness of a given template will peak when

$$I(s;s_1, \ldots, s_m) - I(s;s_1, \ldots, s_{m-1}) \sim 2^m/N.$$

Notice that the second term of the fitness function also provides a mechanism for choosing between two templates of different length and equal I.

We have applied the learning algorithm to two different types of training data. The first test data set was generated by an eight-nearest-neighbor, one-time-step CA rule. The neighborhood of this rule is subsumed by our twenty-site, two-time-step master template. The genetic algorithm found the appropriate template and reproduced the patterns exactly.

The learning algorithm was also applied to patterns

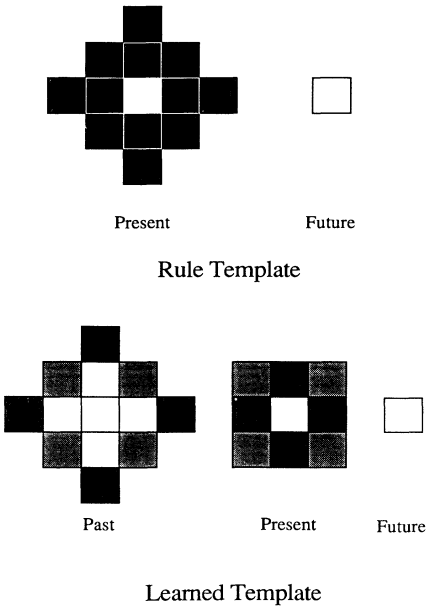

FIG. 2. The sites used (black squares) to create the test data. Note that not all of the sites used by the cellular-automaton rule are available to the learning algorithm. The cellular-automaton rule used the twelve nearest neighbors. The learned template consists of the four second-nearest neighbors of the present, four third-nearest neighbors of the past, and some combination of four second-nearest neighbors from either the present or the past such that no spatial position is selected twice (i.e., the temporal projection of any learned template is identical to the template used by the cellular-automaton rule).

generated by a twelve-nearest-neighbor, one-time-step CA rule (Fig. 2). In this case the task of the learning algorithm is more difficult; the learned rule can only be an approximation of the CA rule, since the learning template does not include the entire template used for generation of the pattern. This type of missing information will be typical of patterns produced in real data where there exist physical variables to which the experimenter has no observational access. Successive applications of the genetic algorithm to these data produced a *set* of optimal templates, each member of which was roughly equally fit.[19] The learned templates are shown in Fig. 2. Each template included all eight of the sites shaded black and four of the eight gray-shaded sites, chosen such that the same spatial position was not selected in both the past and the present (i.e., the temporal projection of any learned template was identical to the template used by the CA rule). In both simulations, approximately 1.5×10^5 data points were used.

In Fig. 3 we compare the results of the learned rule to those of the CA rule used to generate the training patterns. Both rules start with the same two-time-step "seed" and both are iterated independently for 10 time steps. Although not exact, the regenerated pattern captures the main characteristics of the training set. When compared directly to patterns produced by the original rule over a single time step, we find that our learned rule produces correct behavior about 96% of the time, and the template contained about 92% of the total available information.

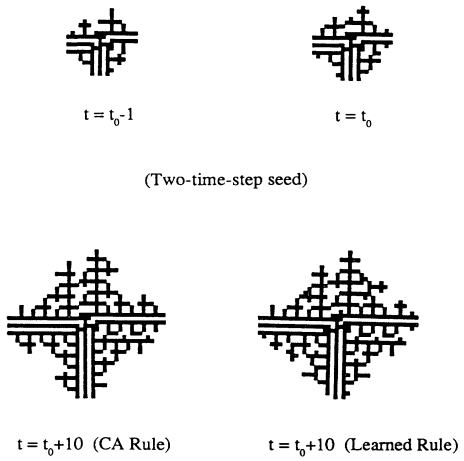

FIG. 3. The top two patterns represent a two-time-step seed. The pattern on the lower left results when the CA rule is iterated for 10 time steps. The pattern on the lower right is the result of 10 successive applications of the rule selected by the genetic algorithm as most fit. Both rules begin with the same seed pattern and then evolve independently.

The efficiency and scope of the learning algorithm may be increased in two ways. First, storing the probabilistic CA rule in the form of a histogram as we have done is inefficient if the table is sparsely filled. In such cases the rule is best stored in the form of a tree data structure. Another generalization of the present approach is to allow the learning algorithm to explore variable spatial and temporal resolutions for the experimental data (we have presented the analysis for a fixed choice of spatial and temporal resolution). The data may be converted to a pyramid data structure[20] that contains the original data as well as successive averages over length and time scales, so that the template search can take place to select the most relevant scales as well as the most relevant spatial orientation.

The question remains: What is the physical meaning of a rule if one can be learned? The simplest interpretation of a "good" rule is that at the particular spatial and temporal resolution embodied in the learned rule, a simple description of the dynamics exists. One might suspect the simplicity of the learned rule to be reflected in phenomenological intuition, but such intuition may be apparent only after the rule is constructed, and indeed, may not be apparent at all.

A more particular question is, how may such a rule be related to theoretical models? If a good rule is learned, then there is an indication that there should be a reduc-

tion from any continuum model to a simpler discrete model. On the other hand, it is possible to use the same techniques outlined here to search larger spaces of models that include variables that are not directly accessible from the data. In the case of solidification, it is certainly clear that the binary-state data exclude many physically relevant variables, e.g., the temperature field. One can, nevertheless, search through simple models that include continuum quantities such as temperature.[21,22] Such models are typically characterized by a set of continuous parameters (e.g., coefficients for terms involving spatial derivatives). In such cases, the map-fitting procedures used in the time-series learning algorithms must be generalized to the fitting of local spatial maps to approximate a partial differential equation.[1] Map fitting, must, however, be augmented by a learning algorithm to search for the "relevant" spatial templates, such as the genetic algorithm methods we present here. Though computationally more demanding, learning in these enlarged spaces could provide solid links between data and theory.

We appreciate helpful conversations with S. Omohundro, R. Shaw, D. Farmer, and J. Crutchfield. N.P. also appreciates helpful conversations at the Aspen Center for Physics session on complex dynamics. This research was supported by Grants No. PHY86-58062-PYI from the National Science Foundation and No. N00014-88-K-0293 from the Office of Naval Research.

---

[a]Inet addresses: [meyer,n,fcr]@complex.ccsr.uiuc.edu; usenet addresses: [ihnp4,ucbvax]!uiucdcs!complex![meyer,n,fcr].

[1]J. P. Crutchfield and Bruce S. McNamara, Complex Systems **3**, 417–452 (1987).

[2]J. P. Crutchfield, Physica (Amsterdam) **10D**, 229–245 (1984).

[3]J. P. Crutchfield and K. Kaneko, in *Directions in Chaos,* edited by Hao Bai-lin (World Scientific, Singapore, 1987).

[4]J. A. Vastano and H. L. Swinney, Phys. Rev. Lett. **60**, 1773–1776 (1988).

[5]S. Ciliberto and J. P. Gollub, Phys. Rev. Lett. **52**, 922–925 (1984).

[6]Y. Sawada, A. Dougherty, and J. P. Gollub, Phys. Rev. Lett. **56**, 1260–1263 (1986).

[7]J. D. Farmer and J. J. Sidorowich, Phys. Rev. Lett. **59**, 845–848 (1987).

[8]J. D. Farmer and J. J. Sidorowich, Los Alamos National Laboratory Report No. LA-UR-88-901 (to be published).

[9]P. Grassberger, University of Wuppertal Report No. WU-B-87-8, 1987 (unpublished).

[10]A. J. Cremers and A. Hübler, Z. Naturforsch. **42a**, 797 (1987).

[11]A method used by S. Ciliberto and M. A. Rubio [Phys. Rev. Lett. **60**, 286 (1988)] for one-dimensional spatial fluid data entails measuring the average noise power over a short time window at a particular lattice site to determine whether the flow is laminar (zero) or turbulent (one).

[12]J. von Neumann, in *Theory of Self-Reproducing Automata,* edited by A. W. Burks (University of Illinois Press, Urbana, 1966).

[13]C. E. Shannon and W. Weaver, *The Mathematical Theory of Communication* (University of Illinois Press, Urbana, 1962).

[14]R. S. Shaw, *The Dripping Faucet as a Model Chaotic System* (Ariel Press, Santa Cruz, CA, 1984).

[15]A. M. Fraser and H. L. Swinney, Phys. Rev. A **33**, 1134–1140 (1986).

[16]A. M. Fraser, "Information and Entropy in Strange Attractors" (to be published).

[17]J. Holland, *Adaptation in Natural and Artificial Systems* (Univ. of Michigan Press, Ann Arbor, 1975).

[18]The reader will notice that the factor limiting the size of our template is strictly empirical. One might be tempted to demand that the fitness function favor a smaller template on the grounds that it is less complex and hence desirable. There is, however, no *a priori* reason to believe that the representation we have chosen for our rules is the most efficient and that a longer template could not be translated into a more efficient rule than could a shorter template. Incorporating a measure of algorithm complexity in a fitness function might only bias it to a specific representation space.

[19]Because of the symmetry of the CA rule used to generate these data, there are several maxima of equal size in the fitness landscape. This enabled the genetic algorithm search to work equally well without the use of crossovers.

[20]P. J. Burt, Computer Graphics and Image Processing, **16**, 20 (1981).

[21]N. H. Packard, in *Science on Form,* edited by S. Ishizaka, Y. Kato, R. Takaki, and J. Toriwaki (Kluwer Academic, Hingham, MA, 1986).

[22]F. Family, D. E. Platt, and T. Vicsek, "Deterministic Growth Model of Pattern Formation in Dendritic Solidification," Emory University report, 1987 (to be published).