Low-Dimensional Dynamics in Agent-Based Models

Nathaniel Osgood

Department of Computer Science, University of Saskatchewan

110 Science Place, Saskatoon, SK S7N 5C9 Canada

306-966-6102 (Office), 306-966-4884 (Fax)

osgood@cs.usask.ca

Abstract

Within recent years, agent-based models have achieved growing prominence in several fields of study. Although powerful and expressive for characterizing the evolution of large populations exhibiting persistent interactions between individuals and high heterogeneity, agent-based methods do not come without tradeoffs. Such methods are burdened by relatively high runtime, lack a formal canonical, declarative, and transparent mathematical semantics, and are often challenging to program, understand, calibrate, generalize and validate. It is therefore important to help modelers recognize modeling contexts requiring the full generality of such models. This paper takes a preliminary step in that direction. Specifically, we built and apply a framework that applies the theory of delay embedding and generic algorithms for intrinsic dimensionality assessment in order to estimate the intrinsic dimensionality of the trajectory of agent-based models. This dimensionality provides a lower bound on the number of state variables required in any model that seeks to reproduce the behavior of these agent based models. Although results are tentative and particularly sensitive to noise, our work appears to indicate that many highly descriptively complex agent-based models may give rise to exceptionally low dimensional global behavior. We suggest that there may be opportunities for expressing the behavior of many complex agent-based models using state equation models offering much far smaller size and greater computational economy.

1 Introduction

Recent years have witnessed an upsurge of interest in agent-based modeling. The interest has been particularly pronounced within fields studying the effects of persistent interactions between individuals, such as the modeling of infectious diseases and the spread of ideologies, but such models have also commanded attention in physiological modeling, marketing, and other areas. For systems containing large populations in which the behavior of individuals is better understood than the emergent behavior of the populations, agent-based models provide an important and expressive tool for understanding the roots of global behavior. The virtues of agent-based models for accurately capturing the effects of population heterogeneity are also important [1].

The benefits of agent-based modeling do not come without significant tradeoffs. The simulation of agent-based models imposes an exceptionally heavy computational burden. The large and distributed state within such models complicates understanding of model behavior. The absence

of a general but precise mathematical foundation for agent-based models greatly limits reasoning about such models, and hinders the introduction of more general approaches to model analysis, validation, and calibration. Even if a consistent mathematical framework is imposed for a particular model, the large number of parameters of such models typically makes the calibration problem underdetermined; as a result, the field still lacks effective general-purpose techniques for calibrating agent-based models. Finally, despite some significant recent advances in introducing limited declarative techniques [2], most agent-based models rely heavily on general-purpose computer languages, which lack basic support for domain-specific semantics such as unit checks and tend to obscure the modeling program with a welter of implementation-specific detail [3]. The result is that agent-based models are typically less nimble and transparent than those built with modern aggregate modeling tools. In infectious disease epidemiology, the complexity and rigidity of agent-based modeling has apparently had significant constraining effects on the development of more detailed models of disease spread.

Given these tradeoffs, practically minded modelers desire understanding to inform basic modeling choices – Under what conditions should one modeling approach be used? Are there times when using several approaches make sense? Are there clear indications as to when one approach is necessary, or when an approach cannot be used?

Several recent contributions have explicitly compared the tradeoffs between agent-based and ordinary differential equation (ODE) based methods [1, 4] and mixing assumptions in network models [5]. In the epidemiology area – where for years compartment models and agent-based approaches have competed – there is growing recognition of the importance of model pluralism [6-8]. [4] demonstrated that aggregate SIR models could accurately capture the dynamics of agent-based network models of disease spread for a wide range of parameter values. While effective calibration of the aggregate models often requires execution of an agent-based model, the modeling process can be more flexible and informative while working with the more aggregate model either following or concurrently [9] with a more focused agent-based model. [1] examined the performance and accuracy scaling of agent-based and attribute-based disaggregation models as the need to represent population heterogeneity rises. That work demonstrated that as the number of heterogeneity dimensions rise, agent-based disaggregation affords faster execution and the ability to representing the dynamics with greater precision.

This paper continues in the theme of these earlier results in seeking to shed light on the tradeoffs between agent-based and ODE-based modeling. Firstly, we describe a mathematical approach we have implemented for studying the emergent, high-level behavior of models. This framework is designed to permit the analyst to look beyond the details of a model's implementation and to examine instead the intrinsic characteristics of its behavior. Specifically, by looking at the intrinsic dimensionality of the model's behavior, we can gain insight into how economically the model describes its behavior, and highlight potential opportunities for representing that behavior in a simpler fashion. Secondly, we describe the application of this framework to several example model results. In so doing, we have discovered that several descriptively complex agent-based models exhibit exceptionally "simple" (low-dimensional) deterministic aggregate behavior, mixed with some stochastics at a local level. While these results are as yet limited to a few examples, they suggest that the global behavior of many systems whose descriptive complexity might appear to require representation as an agent-based model may admit to extremely high-fidelity expression as low-dimensional systems of ordinary or stochastic differential equations. In light of these results, we conjecture that the success of [4]

in identifying a system of ODEs for accurately describing an agent-based model may have broad applicability, and perhaps be even more than rule than the exception.

A few words about the results presented herein. We believe that this is the first time that these techniques (based on dynamical systems theory) have been specifically aimed at analyzing the complexity of agent-based models, and systems simulation models more generally. We believe that there are many opportunities for extending and improving upon the early results presented here. While we have taken one approach to dimensionality estimation, we believe that there are ripe opportunities for application of other approaches. Another concern is the non-constructive nature of our approach: While our technique can demonstrate that it is possible for a given model or system to be represented in a simpler fashion, it does not indicate the specifics of that representation. We believe that there are favorable prospects for applying model order reduction approaches to derive the lower-dimensional models. A number of additional caveats and suggestions for innovation are given in Section 4.5.

The next section of this paper introduces the mathematical background for the two major elements of our approach. Section 3 describes the specific manner in which our approach combines these two elements to estimate the intrinsic dimensionality of model behavior. Section 4 provides case studies in which we estimate the intrinsic dimensionality of several example agent-based models. Section 4.5 provides some high level comments, and discusses future directions for research.

2 Background

This section describes the mathematical background for the three key elements of our approach. Firstly, Section 2.1 describes the notion of the trajectory of a process in state space, and discusses how one of its characteristics – its dimensionality – can provide a lower bound on the complexity of models required to realize that process. Secondly, given that most state spaces cannot be directly observed, Section 2.2 discusses how it is possible to reconstruct the state space of a highly coupled system using time series information from just a single measurable quantity, even in the presence of measurement error. Finally, Section 2.3 describes how we apply existing generic data set dimensionality algorithms in order to estimate the dimensionality of the reconstructed state space, without the need to completely reconstruct that state space.

2.1 State Space Approaches: Concepts and Dimensionality

2.1.1 State Space Representation

Regardless of its implementation, a process can in general be characterized as the time evolution of some state vector $\mathbf{x(t)}$. For a particular time t_0, $\mathbf{x(t_0)}$ completely characterizes the state of the system. In a purely deterministic system, the value of $\mathbf{x(t_0)}$ uniquely determines the subsequent values of $\mathbf{x(t)}$.

While it is most common to characterize the evolution of each component of the state vector explicitly over time (such as is shown in Figure 2), an alternative representation characterizes the evolution as a curve (trajectory) in *state space* (also known as *phase space*). The state space shares the same dimensionality as the state vector (i.e. $\mathbf{x(t)} \in \Re^n$) and represents the set of all possible state vectors. Within the representation of a process as a trajectory in state space, time is implicit rather than explicit. Because a purely deterministic system's state completely

specifies the subsequent trajectory of the system, a closed trajectory will indicate a system with periodic dynamics.

In order to illustrate these concepts, Figure 1 depicts an epidemiological model of a disease conferring temporary immunity. Within this model, individuals can either be susceptible (S), infectious (I) or temporarily immune (TI1). Figure 2 shows the time behavior of the state variables of that model. Figure 3 shows[1] the evolution of those state variables in within state space ($\Re^3$).

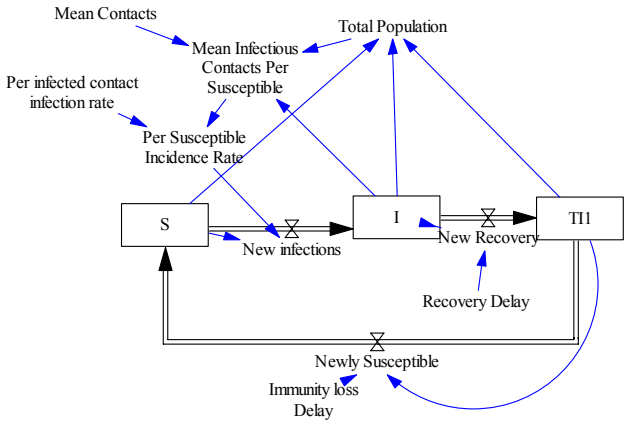

Figure 1: An Epidemiological Model of a Disease Conferring Temporary Immunity

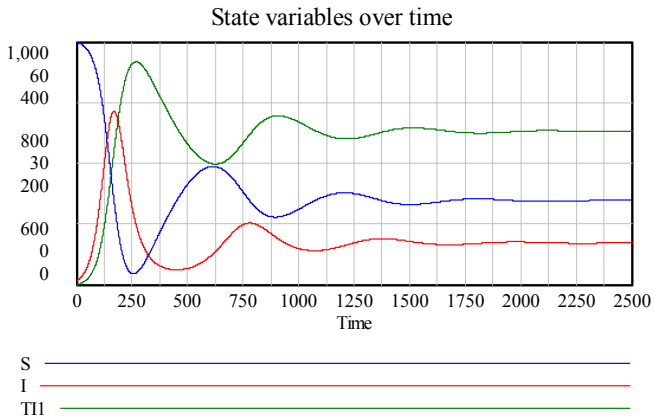

Figure 2: Time Evolution of State Variables

[1] It is in general is only possible to directly depict the entirety of state space for $n \leq 3$.

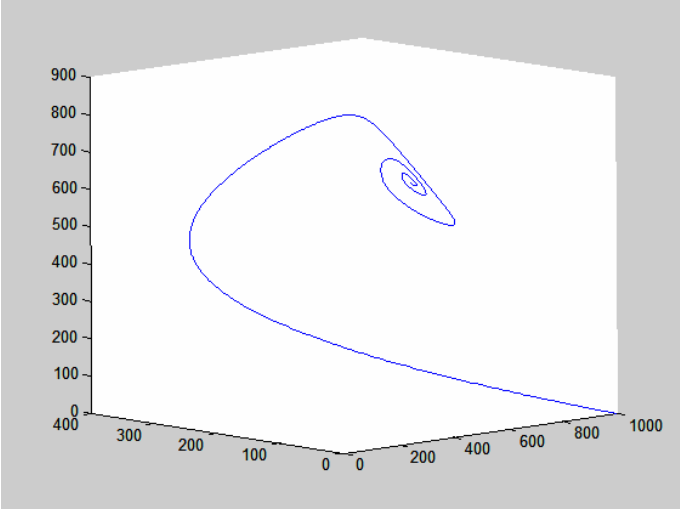

Figure 3: State Space of a Damped Oscillator

2.1.2 Intrinsic Dimensionality of a Process

Although the trajectory traverses the state space, it is not necessarily the case that the trajectory will have identical dimensionality to the state space. In the case above, although the state space is 3 dimensional, it can be readily noticed that the trajectory dictated by the model parameterization given is only 2 dimensional (i.e. only has an *intrinsic* dimensionality of 2). In this particular case, a particularly unusual condition holds: The two dimensional trajectory is co-planar (falls entirely on a single plan), as can be clearly seen by from a view of the state space such as that shown in Figure 4. The co-planarity in this case reflects the fact that there is a conservation of people in the model, and that any of the variables can be expressed as a linear combination of the other two. In other cases, we may have a surface of lower intrinsic dimension (a *manifold*) than the surrounding state space even absent a linear dependence among the state variables.

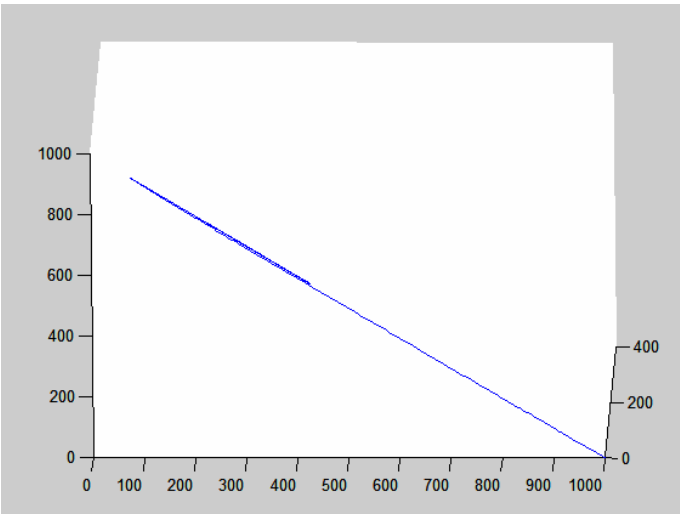

Figure 4: View of State Space indicating coplanarity of trajectory

The intrinsic dimensionality of a process' trajectory is of great interest to modelers, because it is an underlying characteristic of the process itself, rather than an artifact of our modeling choices –

that is, the particular coordinate system that we happened to adopt when characterizing the system. Regardless of the dimensionality of the state space in which it is contained (or the technology by which it is realized), the intrinsic dimensionality of the trajectory indicates the number of independent state variables that would be needed in a model to accurately characterize capture its dynamics. This dimensionality is certainly a lower bound on the dimensionality of the underlying model system[2]. It is possible that a trajectory started with different initial conditions may be associated with different trajectories, but in many cases (such as many ergodic systems [10]), the dimensionality of the trajectory will be equal to that of the model system as a whole.

If intrinsic dimensionality of a process is low, it suggests that a clever choice of a small number of state variables could completely express the time evolution of the system. By contrast, if the intrinsic dimensionality of a process is high, it necessitates a high-dimensional model for expressing that behavior with any fidelity. Emergent behavior of some population that exhibits very high intrinsic dimensionality could motivate the use of agent-based models for describing the evolution of that population.

2.1.3 Execution of a Model as Process

The section above introduced the concepts of state space and intrinsic dimensionality, and the implication of intrinsic dimensionality for representation of a process by a model. This subsection extends the above ideas with respect to a particular type of artificial process: A particular run of a simulation model.

Consider an explicit simulation model. Regardless of whether this model is realized in a general-purpose programming language, a system dynamics package, an agent-based framework, or any other number of frameworks, that model is associated with some state space. The dimensionality of this state space is given by the count of state variables in the model. For a "pure" system dynamics model (i.e. a model without any fixed delays, pseudo random number generators, etc.), the number of state variables is simply equal to the number of stocks. For an agent-based model, the number of state variables will typically be very large – equal to the sum of the number of state variables (dynamic attributes) in each agent over all agent populations, plus any additional state variables used to e.g. simulate the evolution of the agents' environment. For a typical agent-based model, the number of state variables will frequently be in the hundreds, and sometimes significantly orders of magnitude higher.

A particular execution of the model is a process associated with some trajectory through the model state space. If this process is associated with an intrinsic dimensionality smaller than the dimensionality of the state space (as we saw in the example shown in Figure 4), it suggests that there may be some measure of redundancy in the model. More specifically, the gap in dimensionality suggests that the particular parameter settings used, the values of some of the state variables can be expressed as a function of the other state variables[3] much as coordinate

---

[2] It bears noting here that we speak here of the "model system" in an applied mathematics sense, denoting not just the structure and equations, but also the particular parameter values employed.

[3] These minimal state variables could be the same state variables as currently exist in the model, or some other basis set of minimal state variables.

shifts identified using principal or independent components methods can permit a lower dimensional means of describing a seemingly high dimensional statistical data set.

While the state space dimensionality is an artifact of the particular modeling approach, of greater interest here is the intrinsic dimensionality of a state trajectory for this simulation model. The intrinsic dimensionality of a state trajectory of a purely deterministic must be equal to or less than the number of state variables within the model (for it to exceed this limit would require some additional state to be maintained outside the model).

Consider a simulation model associated with state space dimensionality D_S and a state trajectory for that model with intrinsic dimensionality D_I. If $D_S \approx D_I$, it suggests that, informally speaking, the implemented model fully exercises the complexity afforded by its size and that no "simpler" model could adequately capture the dynamics of this model. By contrast, a case where $D_S << D_I$ suggests that the model at hand *may be* (but not necessarily *is*) too complex, and that a substantially simpler model may be able to capture some or all of the dynamics expressed by the generated trajectory – and potentially all of the behavior possibly generated by this complex model for other initial conditions, or even different parameters settings.

Within this paper, we present a means of estimating of the intrinsic dimensionality of trajectories produced by agent-based models, and use that approach to analyze a few particular examples. But prior to so doing, we need to formulate some means of assessing this dimensionality.

2.2 Time Series and State Space Linkage and Embedding

The previous section has made a case for the value of understanding the state space characteristics of processes – particularly their *intrinsic dimensionality* – in order to gain insight into the complexity of models required to adequately represent those processes.

Unfortunately, it is not obvious how to study the characteristics of state space for either natural systems or computer models. Firstly, for natural systems, we frequently have no direct means of measuring the state of the system over time – many components of the state may be unobservable. Secondly, even for artificial systems in which we know all of the state variables, the curse of dimensionality restricts us from trying to directly represent the entire state space of a large ODE model or – even worse – a medium- or large-scale agent-based model. Naïve representation would require infeasibly large amounts of memory.

While more sophisticated approaches are possible, this section introduces an approach for reconstructing the characteristics of a process' trajectory through state space that can be applied to either natural systems or formal mathematical models, and which does not depend on either full measurability of the state vector or explicit representation of the entire state space. Specifically, we describe a known approach for reconstructing the coordinates of points in the state space trajectory of a process using periodically sampled time series data from a single observable quantity determined by the system.

Within the first subsection, we review the basic intuitions underlying the seemingly puzzling connection between lagged samples and state space. The second subsection briefly describes Taken's Embedding Theorem, which formalizes this linkage, and describes a general technique for reconstructing the state space of the original system using sampled data.

2.2.1 Basic Intuitions

Most complex systems exhibit some degree of coupling between many components of the state vector. For some complex systems (including those characterized by many system dynamics models), there is pronounced coupling in the time evolution of different components (stocks). As a result, the evolution of one component of the state vector is likely to be impacted by the values of many other components.

Suppose that we have two different intervals of time of equal duration $I_A = [t_A, t_A+k\Delta t]$ and $I_B=[t_B, t_B+k\Delta t]$ in which a particular component of the state vector $\mathbf{x}\downarrow_i(t)$ is known to take on corresponding successive values over time (i.e. $\mathbf{x}\downarrow_i(t_A+j\Delta t)=\mathbf{x}\downarrow_i(t_B+j\Delta t)$ for $0\leq j\leq k$**.** If $\mathbf{x}(t)$ is generated by a highly coupled system, it suggests that not only is this particular component of the state vector similar for these sequence, but the other values of the state vector are likely to be similar as well. Were the state to differ in these other components during intervals I_A and I_B, we would expect these other components of the state vector to "throw off" the value of component *i* during that time as well – eventually, the coupling would allow these differences to manifest in the value of component *i*. The longer the sequences that are identical with respect to component *i* (i.e. the larger the values of *k*), the more confident we would be that the underlying states are similar during intervals I_A and I_B.

Phrased another way, if we wanted to predict the future values of component *i* based on its past values of that component, within a system of bounded dimension, we would expect that we could do so using only a finite memory. The higher the dimensionality of the system, the greater the number of past values of component *i* we would have to match in order to have confidence of the next value of that component.

While strictly informal, these intuitions turn out to be quite correct – in a highly coupled system, the time evolution of just one component can give us hints regarding the structure of the underlying (and frequently unobservable) state space. The next section discusses the formalization of this idea in Taken's Embedding Theorem.

2.2.2 Taken's Embedding Theorem

In a celebrated result, Mathematician Floris Takens demonstrated in 1981 [11]that it is possible to reconstruct the state space of a dynamical system using only time series data drawn from sampling a single quantity determined by the system. (This work was anticipated by other work [12])

A number of subsequent embedding theorems have sharpened and broadened Taken's Embedding Theorem, and clarified the degree of reconstruction that is possible [10].

The essentials of phase space reconstruction are surprisingly simple and general. Given a (frequently unobservable) state vector $\mathbf{x}(t)\in\Re^n$, we can define a periodic[4] sampling of the state vector as follows:

$$s_i=s(\mathbf{x}(t_0+i\Delta t))+\eta_i$$

[4] The theorems remain robust under many non-uniform sampling conditions as well, but both for simplicity and because of the widespread use of periodic sampling, we focus here on that case.

where s(**x**(t)) simply represents the (scalar) value of some observable quantity dictated by the state, η_i represents some noise associated with each measurement, and Δt represents the lag between subsequent measurements. For the moment, we will assume that the measurement noise is very small; discussion in subsequent sections will indicate how such noise can be recognized and handled.

We can then define *delay reconstruction* vectors $\mathbf{s}_i$ of length D_E as follows:

$$\mathbf{s}_i = \begin{bmatrix} s_i \\ s_{i+1} \\ \vdots \\ s_{i+D_E-1} \end{bmatrix}$$

Just as $\mathbf{x}(t) \in \Re^n$ traced out a frequently unobservable trajectory in state space of embedding dimension $D_S=n$, we can think of the vectors $\mathbf{s}_i$ as etching out a trajectory in a reconstructed state space of size D_E.

While the connections between these state spaces are not at first blush obvious, Takens demonstrated a profound connection between them. In particular, as long as D_E is sufficiently large (at least twice the intrinsic dimensionality[5] of $\mathbf{x}(t)$) and certain other basic but generally non-onerous restrictions are met, there will exist a uniquely invertible smooth map (a diffeomorphism) between the state space of $\mathbf{x}(t)$ and that of $\mathbf{s}_i$. Among other virtues, this map preserves the intrinsic dimensionality of the state space trajectory.

It is important to stress that the requirements on reconstruction dimension D_E are a function not of the dimensionality of the original state space D_S but of the *intrinsic dimension* of the underlying trajectory $\mathbf{x}(t)$. Moreover, in practice, this smooth mapping is often realized for considerably smaller values of D_E. Thus, while it is not in general possible to reconstruct the state space of $\mathbf{x}(t)$ in $\Re^n$ for larger models, in some cases the intrinsic dimensionality is sufficiently small that we can explicitly construct or depict the reconstructed state space. While there can be benefits from such a reconstruction, our interest in this paper lies in assessing one particular characteristic of the reconstructed space: Its intrinsic dimensionality. Because of the similar geometric structure of the trajectory in the two spaces, an estimation of the intrinsic dimensionality of the reconstructed state space will serve as an estimate of the intrinsic dimensionality of the original state space.

The next section looks at particular means for estimating the intrinsic dimensionality of the reconstructed state space.

2.3 Techniques for Dimensionality Estimation

The embedding theorems demonstrate that we can in principle reconstruct the state space trajectory of a complex process using just data drawn from periodically sampled time series data for a single quantity determined by that process. This section focuses on the task of estimating the intrinsic dimensionality of the original state space from the sampled data. For this purpose,

[5] In the sense of "box counting" dimension – a measure that is in practice similar to the correlation dimension discussed below.

we can make use of dimensionality estimation techniques for generic multidimensional data sets. The subsections below review particular approaches that have been used by past work. While none of this work has examined intrinsic dimensionality estimation in the particular context of simulation model output, the approaches apply directly to this novel context.

While embedding theory provides us with the necessary tools to reconstruct an explicit representation of the trajectory within state space, it is fortunate that an estimation of intrinsic dimensionality does not require us to do this. Instead it will turn out that it is sufficient to simply measure distances between the points in that embedded space, without the need for any persistent representation of all of the points on the trajectory. But in order to understand this procedure, we will have to present the basic mechanisms in a manner that will make reference to the set of embedded points. The reader should keep in mind that this is a conceptual rather than computational construct.

2.3.1 Epistemic Limitation

The previous section described how we could use delay embedding for a time series to produce a set of points S in some space of dimensionality D_E. The points of S are the delay reconstruction vectors $\mathbf{s}_i$, and each element in those vectors is a particular measured sample from the system.

It is worth emphasizing that when we begin this process, we lack knowledge of the "proper" dimension for the delay reconstruction vectors. This is significant, because it imposes a limitation on our estimation procedure – In order to capture scaling behavior with exponent d, we need to be using delay reconstruction vectors of least dimension $2d$. On the other hand, assuming too high a dimension for our delay reconstruction vectors has relatively little cost. It is thus prudent to use a conservatively high dimension for the vectors. In the rest of the paper, we will term this tentatively chosen embedding dimension D_E, in order to distinguish it from the hypothesized intrinsic dimension D_I.

Given such a set S, we can then apply a number of generic techniques for estimating, the intrinsic dimensionality of a set of points. This section discusses those dimensionality estimation techniques.

2.3.2 Dimensionality Measures

Given some trajectory T in $\Re^n$, there are many possible approaches to defining – much less measuring – the dimensionality of T. Examples include the Hausdorff-Besikovich (or "fractal") dimension [13], the "box counting" dimension [10], the "information" dimension [13], and the correlation dimension (first proposed in [14]).

The dominant method of dimensionality estimation is based on the *correlation dimension*. It can be shown that the correlation dimension is a lower bound to the more precise "box counting dimension", but it is believed that the difference between the two is very small – and perhaps negligible – for many practical examples [15]. In contrast to some of the other measures, the correlation dimension is also simple to define and relatively efficient to compute.

The intuition behind the approach is as follows: Suppose we are given a set of points S, each of whose members is an element of $\Re^n$. Regardless of the value of n, if we are working with a structure with d intrinsic dimensions around some point, we would expect the number of points lying within a hypersphere of radius r around that point to increase as r^d. We would expect the

distances between pairs of points drawn from S to scale similarly. In other words, we would expect

$$\sum_{p \in S} \sum_{q \in S, p \neq q} I(|p - q| \leq r) \propto r^d$$

where I is the indicator function returning 1 if the argument is true and 0 otherwise.

In order to estimate the correlation dimensionality of S, we therefore examine how the cumulative empirical probability distribution of the distance between pairs of points within S scales with r. More specifically, for a fixed set S we define

$$C(r) = \frac{\sum_{p \in S} \sum_{q \in S, p \neq q} I(|p - q| \leq r)}{|S|^2}$$

We further define the correlation dimension as

$$\lim_{r \to 0} \frac{\ln C(r)}{\ln r}$$

As discussed further below, in practice we often need to look at the variation around other regions than r=0, particularly in the context of noise. We will be adopting the correlation dimension in our approach to estimating the dimensionality of the trajectory of a process.

2.3.3 Point Estimation of Correlation Dimension

The subsection above has described the quantity to be calculated $\left(\frac{\ln C(r)}{\ln r} \right)$ but not a specific means of coming up with a point estimate. There are a number of algorithms that have been developed for performing the above estimation. Grassberger and Procaccia's original algorithm [13] appears to have involved simple plotting of *C(r)* vs. *r* on a log-log graph and "eyeballing" the scaling behavior. Specifically, the authors took the slope of the curve (although not necessarily around r=0) to indicate the correlation behavior. [11] proposes a maximum likelihood estimation approach that permits simple calculation, but is forced to deal with randomness as a special case. The robustness of this algorithm to the statistical fluctuations that occur in smaller datasets was also unclear to the author. [15] provide a refinement of Grassberger and Procaccia's algorithm to permit improved estimation for smaller numbers of time series samples. Their results suggest a means of refining the estimates examined in this paper.

Our approach arrives at a point estimate for the correlation by an "eyeball" approach, although our library significantly simplifies the process by directly indicating the slope of the line at different values of *r*. The user must merely choose the appropriate scale (value of *r*) at which to determine the correlation dimension. As discussed below, this is generally determined by looking at the graph of *lnC(r)* vs. *ln(r)* for different embedding dimensions D_E.

2.3.4 Accuracy and Sample Size

Past work [16, 17] has demonstrated that the set S of embedded points must reach a certain level to guarantee a good estimate of C(r). Specifically, to estimate an instrinsic dimension D we

require at least $10^{\frac{D}{2}}$ data points in S (or 10^D *pairs* of data points drawn from S). [15] have demonstrated, however, that appropriate correction can allow data sets not meeting this criterion to produce good results in practice. It bears noting that there is something of a chicken-and-egg problem here, as we do not know the number of points required in the analysis until after the analysis is successful. At the cost of a computational workload, it is therefore safer to err on the side of overestimating the possible dimension.

2.3.5 Computation Demands and Probabilistic Sampling

Because of the need to examine distances for each pair of points, computation of C(r) requires time quadratic in the size of the data set (i.e. is $O\left(\left|S\right|^2\right)$) For large sets S and low intrinsic dimensionality, the exhaustive computation of C(r) may be needlessly expensive. We introduce the approach of probabilistically estimating C(r) by Monte Carlo sampling. Within this paper, we adopt this approach. Specifically, we define $P \in S \times S$ where each element of P is a pair of randomly selected elements of S, and $|P|=N$. We then compute the approximation to C(r):

$$C(r) \approx \frac{\sum_{(p,q)\in P} I(\left|p-q\right| \le r)}{\left|P\right|^2} = \frac{\sum_{(p,q)\in P} I(\left|p-q\right| \le r)}{N}$$

C(r) can be estimated in this manner in a fully general form (i.e. without pre-set fractiles) within time O(NlogN) (with the bottleneck being the time necessary to sort the N distances). If we are willing to simply count a histogram of empirical fractiles with known spacing, we can perform the computation in time O(N).

By changing N, we can trade off running time and accuracy in the estimation of C(r).

2.3.6 Stochastic Effects

To this point, we have treated the underlying state trajectory as deterministic. In practice, the values of the time series will typically incorporate a non-trivial stochastic component. This component may reflect several things, including inherent stochastics or round-off error in the original system or measurement error.

Recall the earlier reasoning concerning the intrinsic dimension of a system and its time series from Section 0: The larger the inherent dimension of the system, the larger the number of samples needed to predict the next output of the system. An ideal stochastic source cannot be characterized by any finite dimension: No matter how many previous measurements we take, we cannot predict the next measurement. As a result, stochastic effects will appear to have *infinite intrinsic dimensionality*.

In practice, we tend to see stochastic effects more at certain scales, and less at others. We can recognize those effects by considering the estimate of log*C(r)* vs. log*r* for several different sizes of delay embedding dimension D_E. For those scales ($r=r_0$) at which deterministic effects dominate, we expect the slope of the graph of log*C(r)* (i.e.

$$\left.\frac{d\ln C(r)}{d\ln r}\right|_{r_0} = \left.\frac{\left(\frac{d\ln C(r)}{dr}\right)}{\left(\frac{d\ln r}{dr}\right)}\right|_{r_0} = \left.\frac{\left(\frac{1}{C(r)}\frac{dC(r)}{dr}\right)}{\left(\frac{1}{r}\right)}\right|_{r_0} = \left.\left(\frac{r}{C(r)}\frac{dC(r)}{dr}\right)\right|_{r_0}$$) to stabilize around the true intrinsic dimension. By contrast, for those scales at which randomness dominates, no matter how large D_E may be, we expect the dimension estimate of to rise. These results are borne out in practice.

2.4 Summary

Within this section, we described background information on several different subpieces of the problem at hand. We introduced the notion of the intrinsic dimensionality of a process, a quantity that is independent of implementation or the particular choice of variables used to realize that process. We observed the fact that one quantity from a highly coupled process is influenced by all elements of the state of the process, and the relationship between a periodically sampled time series of some quantity from a highly coupled process and the dimensionality of the underlying process. We saw that in such a coupled process, such a time series can allow us to reconstruct a close approximation to the trajectory of the process that gave rise to it.

This fact opened the opportunity to apply generic algorithms for estimating the intrinsic dimensionality of a set of points in order to estimate the intrinsic dimensionality of the generating process. In particular, we have introduced a means of determining *correlation dimension* of the delay reconstruction data as a practical means of estimating the intrinsic dimensionality of the original system state space. Given sufficient resources, we can calculate C(r) precisely, or create a probabilistic estimate. Given C(r), there are several techniques for estimating the correlation dimension. Given several initial assignments of embedding dimensions D_E, we can separate random from deterministic effects, and estimate the correlation dimension for those deterministic sections. Within certain parameters (concerning number of data points and embedding dimension used), these algorithms can give estimates of intrinsic dimensionality of guaranteed accuracy. Outside these bounds, the results are often still accurate in practice, although this cannot be guaranteed.

The next section describes the way in which our particular approach integrates these elements to perform a single approach to estimating the dimensionality of simulation models.

3 Summary of Approach

3.1 Dimensionality Estimation

A library of MATLAB functions was created to automate almost all of the process of dimensionality estimation from output data series. Because of the large size of data sets involved, the routines were designed to lower memory footprint by eliminating intermediate results – thereby increasing performance by minimizing the likelihood of virtual memory thrashing.

Initial time series have been imported as MATLAB vectors. The system collects samples of distances between uniformly selected reconstruction vectors. This operation is performed in such a way as to avoid explicitly creating a set of embedded points. Instead, the system randomly selects pairs of embedded reconstruction vectors from a pair of uniformly distributed

points indexed from the timeseries. The code returns an array of distances of user-specified length.

The set of distances so obtained is then normalized by its largest value and sorted in ascending order. The contents of the array give the normalized distances (i.e. *r*), and the normalized index of each in the array indicates the empirical fractile associated with this distance – i.e. *C(r)*. This information is sufficient to construct a graph depicting $\ln C(r)$ vs. $\ln r$, and thereby to construct an eyeball estimate of the correlation dimension (a method used in [13]).

Although the definition of the correlation dimension involves looking at the asymptotic scaling of $\frac{\ln C(r)}{\ln r}$ as r approaches 0, use of eyeball estimation suggests that for practical systems in which we have imperfect measurement or some inherent stochastics (e.g. due to roundoff error), it is helpful to examine the slope of $\ln C(r)$ vs. $\ln r$ at many scales of *r*. Takens noted as early as 1981 [11] that stochastic effects (associated with infinite dimensionality) frequently dominate at smaller scales. In addition, because of the logarithmic scaling of r, statistics for small r tend to exhibit high variability due to low sample size. Because of these effects, it is often the slope of the $\ln C(r)$ vs. $\ln r$ curve at somewhat higher values of r that converges at the intrinsic dimension.

We have found it tedious and error-prone to estimate the slope of the curve at different points. As a result, we designed our library to permit the user the option of taking another step, and allowing the library to computationally estimate the slope of the curve over a wide range of *r*. Performed naively, this process tends to overemphasize small variations in local slope. We have found that some measure of smoothing is highly desirable. As a result, we interpolate $\ln C(r)$ and $\ln r$ in a piecewise linear fashion over an identical fixed grid with 20 points, and then show the ratio of slopes of each within that interval as the estimated correlation dimension for that interval.

By presenting such a graph for multiple embedding dimensions D_E, it is typically possible to clearly distinguish regions dominated by stochastics on the one hand and deterministic effects on the other, and to accurately estimate the correlation dimension for the latter. For example, Figure 5 shows a plot of the slope of $\ln C(r)$ vs. $\ln(r)$ for a synthetic structure of two dimensions for different scales of $\ln(r)$ (abscissa) and different D_E (mapped to distinct colors). It can be readily seen that the estimates of correlation dimension stabilize for r exceeding a particular value (in this case, $r \geq .001$ for the synthetic two dimensional structure and $r \geq .01$ for the synthetic three dimensional structure); by contrast, the dimensionality estimates for r below that threshold expand continuously as D_E increases.

3.2 Interface to Agent-Based Models

This section describes our implementation of a system based on the above ideas for estimating the intrinsic dimensionality of descriptively complex agent-based simulations. All agent-based simulations were carried out using AnySim 5.2 [2]. To enhance familiarity for readers and permit reproducibility of results, most models studied were sample simulations bundled with AnySim. For each model, an output quantity was selected. In all cases, this was an aggregate property of the entire population – for example, the count of individuals infected within a population, or the total size of a particular population.

Each model was modified by adding a timer, along with a single line of event code that periodically reported the value of this global value to the output log in a comma-separated format. This modification had no impact on the system's behavior other than a slightly lower performance. Our early investigations of the intrinsic dimensionality of agent-based models found that the quality of the results degenerated rapidly as the sampling frequency increased; specifically, low sample rates led to failure of dimensionality estimation to converge as D_E increased. The frequency of the timer for agent-based models was therefore set to allow for very fine-grained observation of changes in the output variable. (Table 1 shows for each experiment the settings for these timers as well as other parameters discussed below).

Agent-based models typically required hours of computation and produced 100,000 or more samples of the global value. The results of these simulations were placed in comma separated variable text files and imported into MATLAB. Subsequent analysis took place within MATLAB.

Agent-Based Model	**Reporting object**	**Reporting (timer) period**
Infectious Ants	Number of afflicted ants	1
Predator Prey	Size of hare population	.0005

Table 1: Analyzed Agent-Based Models

4 Results

In this section, we describe the results of our experiments with our dimensionality analysis system.

4.1 Synthetic Structures

In order to validate correct operation of the dimensionality estimation algorithms we estimated the dimensionality of synthetic structures of known dimension. To do this, we created structures of 2 and 3 dimensions (i.e. D_I=2 and D_I=3) in a higher dimensional space of D_E dimensions, with D_E varying between runs. Each point in the structures was created by setting the first D_I elements of a vector of length D_E to random values uniformly distributed between 0 and 1. These vectors were then treated as the results of embedding, and analyzed for intrinsic dimension. 200,000 distances were randomly selected among these structure for determination of C(r). The entire process was repeated for a set of different dimensions D_E. Specifically, the two dimensional structure was examined in each integral embedding dimension from 2 through 5, and the three dimensional structure from 2 through 7. The results of these experiments are shown in Figure 5.

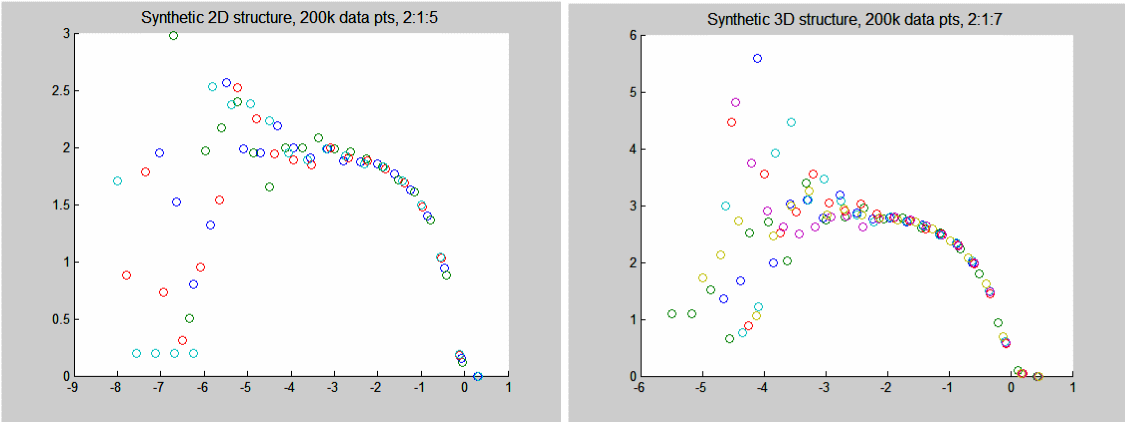

Figure 5: Correlation Dimension vs. ln(*r*) for a Synthetic Structure of 2 (left) and 3 (right) Intrinsic Dimensions.

Figure 5 illustrates the characteristic impact of randomness on graphical estimates of correlation dimension. Because the synthetic structures were composed of the use of randomly selected coordinates for the first several dimensions, stochastic effects dominate for small distances (small values of ln(*r*)). Because true random effects have no true dimensionality bounds and because even pseudo random effects exhibit high intrinsic dimensionality, dimensionality estimates for small r vary strongly with D_E – the higher the embedding dimension, the higher the estimated correlation dimension.

At a wider scale (larger distances), the intrinsic dimensionality of the structure dominates. Recall that the correlation dimension is defined as

$$\lim_{r \to 0} \frac{\ln C(r)}{\ln r}$$

Because stochastic effects dominate for the smallest r, the best estimates of correlation dimension are given by the slope of $\ln C(r)$ vs. $\ln r$ at r= r_d, where r_d is the smallest r for which the algorithm yields convergent results for sampled embedding dimensions D_E above some minimal D_E^*.

Another related characteristic evident in Figure 5 is the impact of the embedding dimension on the intrinsic dimensionality estimates. Geometric constraints prevent us from adequately performing the delay space reconstruction of a trajectory of intrinsic dimension D_I in an embedding dimension $D_E<D_I$. While the approach to dimensionality estimation presented in this paper avoids the need for an explicit reconstruction of state space, the correlation dimension estimates may be poor for $D_E<D_I$. Once D_E reaches or exceeds D_I, these estimates will typically converge to the same value for ranges of r in which deterministic effects dominate. These estimates are guaranteed to converge to D_I for $D_E>2$ D_I, provided that sufficient samples are taken.

4.2 Lorenz Attractor

A second set of validation experiments estimated the intrinsic dimensionality of another, more complex object of known intrinsic dimensionality: The Lorenz Attractor [18]. The correlation dimension of this object is rougly 2.05 [13], and the attractor has served as a benchmark for other dimensionality estimation approaches [13].

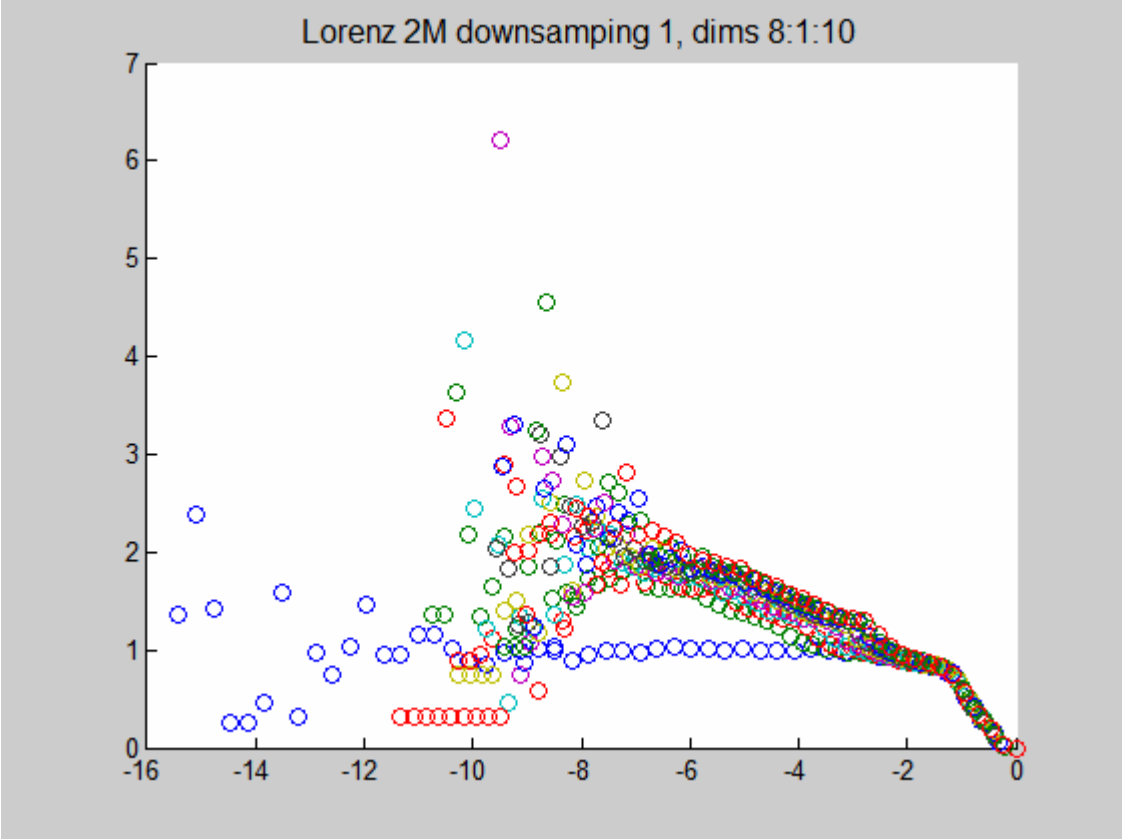

Figure 6: Estimated Correlation Dimension vs. ln(r) for the Lorenz Attractor. Stochastic effects predominate for ln(r) < -2. The varying estimates of the correlation dimension for ln(r) ≥ 2 arise from limitations in accuracy for small embedding dimensions. Estimates converge to a correlation dimension estimate of 2 at ln(r) ≥ -2.

As in Figure 5, random effects predominate for small r (in this case, ln(r)<-2). For ln(r)≥-2, we continue to see some variability in the estimate of the correlation dimension, although it is markedly less than for small values of r. The variability for ln(r)≥-2 reflects the varying quality of the estimate of the correlation dimensional for different embedding dimensions D_E. Empirically, we have noticed that (for deterministic regions) these estimates tend to rise rapidly for $D_E<D_I$, and then converge (although sometimes slowly) to some asymptotic curve giving the correlation dimension vs. r. This observation is borne out for Figure 2, where there is relatively little difference between the correlation dimensionality estimates for D_E=8 and 10.

4.3 Infectious Ants

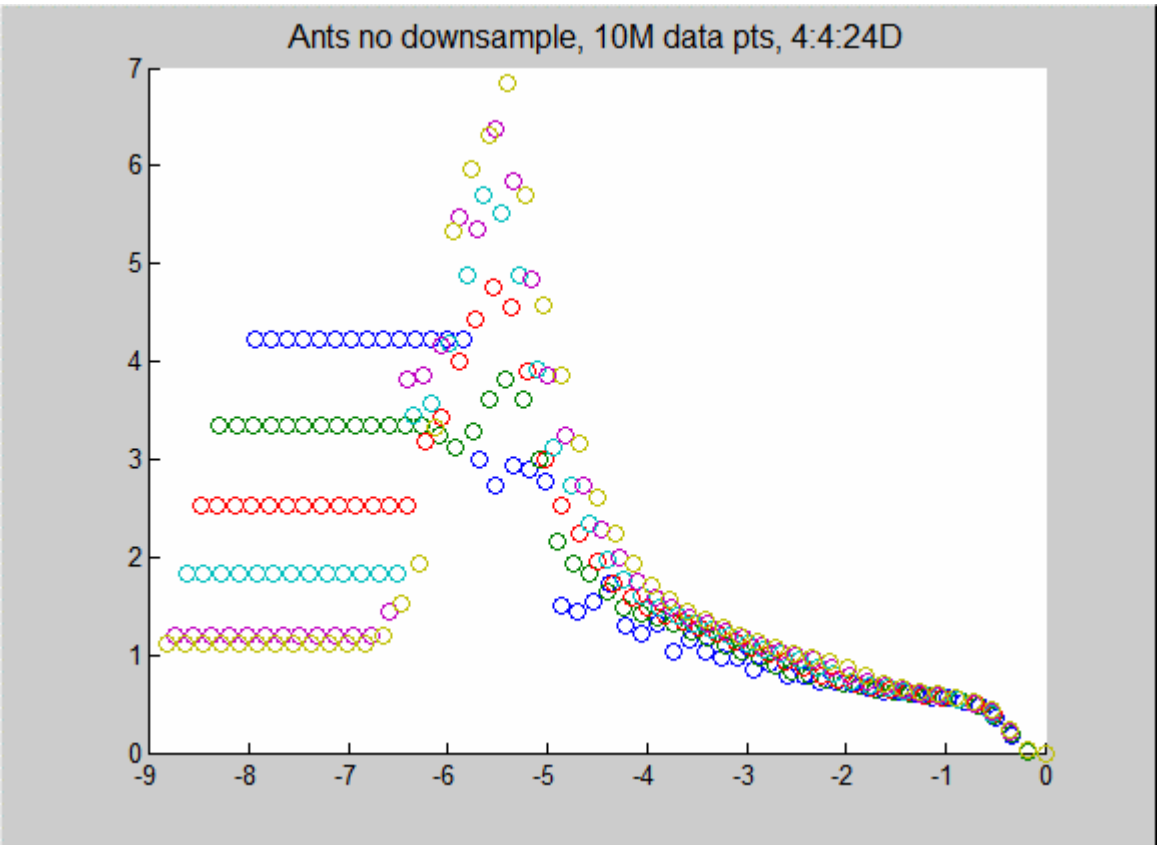

Figure 7: Estimated Correlation Dimension vs. ln(*r*) for Agent-Based Model of Infectious Ants.

Our third experiment investigated the correlation dimension of a richly detailed agent-based model of infection spread. This model (henceforth termed "Infectious Ants") was selected because it includes high degrees of spatial and per-agent (ant) heterogeneity. The model describes the dynamic behavior of a population of 200 ants, each of which is associated with several pieces of state, including x and y location, current direction of travel, health status (healthy or afflicted), and an indication as to whether they are capable of curing another ant (i.e. whether they are a "doctor" ant). Ants wander in a particular direction until they encounter an obstacle (one of the walls) or enter the proximity of another ant. Upon encountering the boundary, ants are reflected in such a manner that their angle of reflection is equal to their angle of incidence. Their behavior in the presence of another ant is more complicated, and involves collision avoidance reasoning. If an infected ant encounters a healthy ant, the healthy ant becomes infected, unless the healthy ant is a "doctor" ant, in which case the infected ant recovers. In the absence of curative treatment by doctor ants, ant recovery times are exponentially distributed (mean duration 100, regardless of further encounters of infected ants that may have occurred since the original infectious encounter). Ants can also develop a spontaneous affliction; the duration of unaffected health is also exponentially distributed (mean duration 2000).

Based on the high dynamic complexity of this game, we expected a high intrinsic dimensionality in the aggregate dynamics. This expectation was not borne out by the results (shown in Figure 7), which suggest an intrinsic dimensionality of approximately 4. The low dimensionality of the

result has proved robust under varying ranges of embedding dimensions D_E, sufficiently high reporting frequencies and sample counts of distances used in dimensionality estimation.

4.4 Predator Prey

As a fourth subject of study, we estimated the intrinsic dimensionality of a particular trajectory of an agent-based model of a spatially disaggregated predator-prey system. This model simulated the evolution of two animal populations, one a predator (lynx) and the other prey (hares). Each animal was associated with a particular age and location in a two dimensional plane tessellated into regular rectangular cells. As in classic models such as Latka-Volterra, lynxes and hares both have the potential for reproduction within a given timestep, with the likelihood of reproduction of the hares exceeding those of the predators. Hares are born into the cell of their parents, unless that cell is overpopulated with hares, in which case the birth is treated as occurring in less populated cells around. Normally 5 hares are born to a hare at once (except for overpopulation). The frequency between births is exponentially distributed with mean duration 4. Lynxes are always born in the same cell as their parents and are born more frequently (mean duration 2) and in smaller numbers (3 per birth) than hares.

Within each timestep, a predator has a likelihood of "catching" prey that depends linearly (up to unity) on the count of prey occupying the same cell as the predator. Predators either consume an adequate supply of prey within each time step or move to an adjacent cell if inadequate prey is unavailable. If a predator is unable to find any prey within some fixed time, the predator dies.

Both predators and prey die of natural causes at a fixed age, with that of lynxes (8) exceeding that of hares (3), if they do not first perish due to other causes. Enforcing these fixed durations requires maintaining an animal's age as a component of the animal's state.

The initial populations of hares and lynxes are 5000 and 40; as in the Latka-Volterra model, the model exhibits non-sinusoidal oscillatory behavior, although for the case of the agent-based model it is quasi-periodic rather than strictly periodic. The model thus represents a trajectory through a sample space whose dimension varies dynamically with population. Based on the fact that each animal is associated with a state consisting of size 3 (age and two location coordinates), we estimate the state space dimensionality of the system experienced during the sampled run to extend well into the thousands.

Because of the high dimensionality of the model state space, we were surprised to discover that the system's trajectory also exhibits very low-dimensional dynamics. As depicted in Figure 8, the estimated intrinsic dimensionality of the trajectory appears to be approximately 5. The estimates of this dimensionality vary for lower D_E but converge for $D_E \geq 10$. The stability of this result was tested for distance sample sizes between 200,000 and 5,000,000.

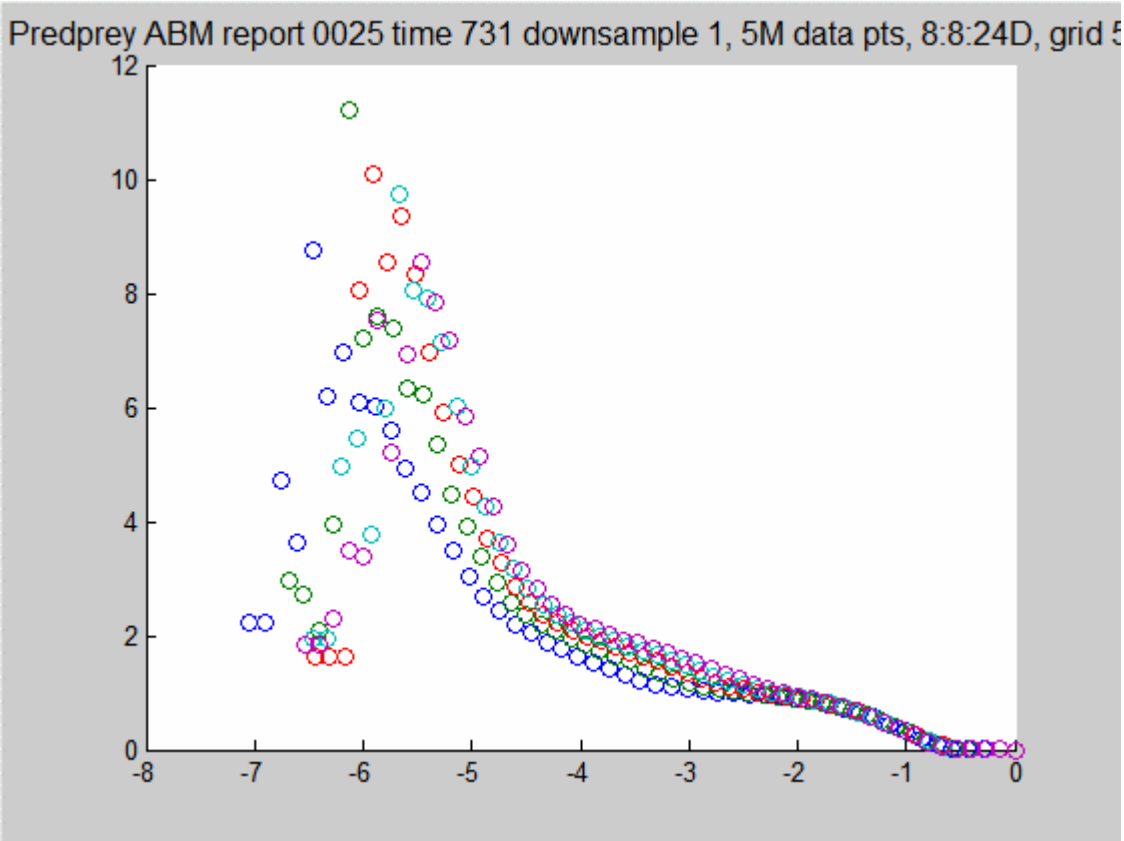

Figure 8: Estimation of Correlation Dimension vs. ln(*r*) for Agent-Based Predator Prey Simulation

5 Discussion

5.1 A Surprise – Or Not?

This paper has presented preliminary experiments investigating the intrinsic dimensionality of the output of descriptively complex simulations. While this work is still in its early phases and is as yet highly tentative in its implications, these results suggest that we may have discovered evidence of very low-dimensional structure in the global behavior of some nominally high-dimensional agent-based model systems. Although the multi-agent systems literature is large and diffuse, we believe that this paper offers the first instance in which the tools of embedding and dimensionality estimation have been used to study simulation output. An earlier paper ([19]) performs a dimensionality analysis of complex model simulation output, but their method employs and much simpler and more limited form of linear analysis.

From a global viewpoint, the recognition of low-dimensional dynamic structure in the emergent behavior of a complex system is hardly unusual. Physicists have long recognized that in the presence of symmetry, conservation laws or simple boundary conditions, the order parameters of highly complex physical systems can exhibit very low-dimensional dynamics [10]. Much of the classic approaches to very high (and frequently infinite) dimensional systems (such as are seen in solid or fluid mechanics) proceed by appeal to such regularities – permitting, for example, the reduction of systems of partial differential equations into sets of ordinary differential equations.

We believe that it may in theory be possible to recognize similar opportunities for model order reduction in the design of simulation models.

We hope that by further investigation of these results, we may learn to better recognize conditions in which the full generality (and attendant complexity) of high-dimensional models is required, and to identify opportunities for reducing model complexity and computational demand through model order reduction.

5.2 Shortcomings

While we believe that the results presented here are of potential significance to significant ranges of complex system models, they do suffer from significant limitations, including the following:

- **Restriction to Periodic Systems.** The dimensionality estimation procedure discussed here is only applicable to systems exhibiting sufficiently dense exploration of regions of state space. Absent repeated visitation to particular regions of state space, a trajectory will appear to have only 1 intrinsic dimension (serving, as it were, as a thread through state space, rather than as a multidimensional manifold in that space). Given the embedding theorem, this would mean at its least that the time series must repeat similar outputs many times. The larger our intrinsic dimensionality, the greater and greater the extent of the sequence that must be repeated in order to properly estimate the dimensionality of the trajectory.
- **Restriction to a Single (Parameter- and Initial-Condition-Specific) Execution.** A second and highly important restriction is related to the fact that the approach estimates the intrinsic dimensionality of a particular model trajectory – that is, a specific execution of the system rather than of all possible executions of the model. Each execution of a model is specific to a particular set of parameter values and initial conditions, and in a general system it is quite possible that the dimensionality of trajectory will differ under different parameter settings or initial conditions[6]. There is thus no guarantee that the particular intrinsic dimensionality estimated is the maximum dimensionality of the model. Indeed, the very notion of "model" differs between dynamical systems theory (where a model is associated with a specific parameterization) and computational modeling (where we frequently speak of a model in terms of its constituent equations/structure, abstracting over the details of the parameter settings). Although we have explored the idea of concatenating sequences of runs of the same model produced by different parameters and initial conditions in order to estimate the intrinsic dimensionality of the model as a whole, we do not feel that this approach is satisfactory.
- **Residual Subjectivity.** Despite the high levels of automation achieved by the library submitted with this paper, the estimate of intrinsic dimensionality still retain some measurement of human judgment in the selection of the time delay between samples, in the decision as to what embedding D_E to examine, the choice of the number of sample points, and in deducing a dimensionality estimate from the interpretation of the graphs. We believe that some measure of this 'art' can be effectively removed by further

[6] It is worth noting that we would expect the dimensionality of all trajectories of an ergodic model to be the same regardless of initial condition.

elaboration of the framework (for example, automated determination of autocorrelation characteristics of the input time series could potentially automate the selection of an appropriate sample spacing), but some element of human judgment is likely to remain.

- **Inability to Exploit Multiple Measures.** One of the great motivations for using embedding theory is the fact that even in the presence of a primarily non-observable system, it is possible to deduce information regarding the structure of the state space trajectory from just a single system output (and even a noisy output). However, even in the context of a largely non-observable system, we often have recourse to several system outputs. It would be desirable to be able to use several outputs to refine the estimate of system dimensionality or the reconstruction of the state space beyond what could be achieved with just one input. Unfortunately, existing theory does not seem to provide an elegant means of addressing this problem. We intend to investigate the effectiveness of simple modifications of the methodology (for example, performing embedding on interwoven time series of pairs, one drawn from each system output), but believe that a more general approach may be possible.
- **Non-Constructive Analysis.** While the revelation that an agent-based model exhibits low intrinsic dimensionality can be encouraging for model developers seeking to characterize complex populations using ordinary differential equations (ODEs), the analysis provide no help in formulating the most appropriate ODE model. While it is always encouraging to know that a low-dimensional model is in fact feasible, it would be desirable to have some assistance in identifying such a model. To address this need, we believe that model developers need to pair dimensionality analysis with other techniques. In particular, we see strong opportunities for the application of model order reduction techniques developed in other fields to simulation models. Constraints on this approach result from the fact that many types of simulation models (notably including agent-based models) as yet lack a well-defined mathematical semantics. In addition to opening the door to application of methods of formal model order reduction, we believe that addressing this shortcoming is a high priority and will yield many additional benefits.

5.3 Future Work

We intend to extend this work by analyzing additional agent-based models, further refining the presented framework. Specifically, we plan to introduce certain analytic simplifications and to improve the precision of the results. We also hope to performance optimizations to improve running time, thereby allowing for analysis of larger sample distance sets. We also plan to explore some of the directions discussed in the previous sections to deepen and simplify the analysis. Given the difficulties handling the correlation dimension with noisy data, we also believe it will be prudent to examine alternative approaches, such as the use of spectral complexity measures. Techniques drawn from dimensional scaling approaches and formal model order reduction also provide attractive avenues of research aimed at lightening the performance burden of complex agent-based simulation models.

A particularly important component our future work reflects our belief (first expressed in [3]) that the lack of explicit mathematical semantics for most agent-based models is a significant liability. In addition to limiting model order reduction approaches, we believe that the lack of an explicit, common mathematical framework for such models also adversely affects reasoning about, generalizing from, validating and calibrating agent-based models. It likely also impedes

the accessibility and modifiability of agent-based models by hindering the introduction of a declarative language for model specification. For this reason, our research group is formulating approaches that can provide compliant, declarative, agent-based models with precise and transparent mathematical meaning.

6 Appendix 1 – Matlab Package Notes

In order to allow for others to benefit from the work described herein, we have made the Matlab source code available for free distribution. The source code is included in the CD-ROM proceedings and is also available by request from the author.

The source code consists of a set of Matlab functions. Many of the functions are helper or wrapper functions concerned with computationally undemanding but convenient tasks such as creating graphs, iterating through common sets of experimental parameters. There are a few functions (indicated for convenience with an asterix) that perform key processing steps, such as deriving distances from the user-specified time series data, estimating $C(\varepsilon)$ from normalized versions of those distances, and calculating the estimated dimensions from the estimated $C(\varepsilon)$.

A few words are in order with regards to the code and specifications below.

- **Naming:** Functions and parameters are given descriptive names, but must live within the constraints imposed by the Matlab restrictions on lengths of identifiers. Names often included components that are variants of the Hungarian variable naming system [Simonyi ref]. This system encodes type (representation) information within names in order to reduce confusion regarding the representation or function of a value held by a variable or passed/returned from a function. Thus, a prefix of "rg" indicates an array, a "str" prefix denotes a string, "ct" a count, etc. In accordance with the convention in the literature on time series embedding, we use the prefix "a" to indicate a time series vector.
- **Optimization opportunities.** While limited amount of Matlab-specific optimization has been performed (e.g. the use of vector operations, pre-allocation of large arrays), the current code is optimized for simplicity rather than speed. In part, this reflects the difficulty of significantly improving code performance within the high overhead imposed by Matlab's code interpretation framework and the desire to maximize portability by avoiding code that would rely upon the presence or require configuration of an external compiler. While we believe that significant gains would be achieved by using a lower-level compiled language for the core computational components of the code, we believe there are greater opportunities for streamlining existing algorithms (for example, the use of an $O(n)$ rather than $O(n\log n)$ estimation procedure for $C(\varepsilon)$) within Matlab without sacrificing portability. We anticipate future versions of the library which take advantage of these opportunities.
- **Specification Notation**. For the sake of brevity and conceptual clarity, we have taken certain liberties in notation below. Rather than using the strict data types associated with symbols, we have indicated the domains modeled by those data types, or by the symbol itself (whichever is more restrictive). For example, $\aleph$ is used to indicate a natural number (non-negative integer), $\Re$ a real, exponents used to indicate arrays/vectors of numbers, etc.

6.1 Analytic Functions

DetermineAndDisplayCorrDimensionFromTimeSeriesForRgEmbDims

Informal description

Creates new graph and displays estimated correlation dimension (derivative of C(ε) for different trial embedding dimensions on that graph, as specified by the array of integral trial dimensions rgDims.

Arguments

aTimeSeries : $\Re^N$	The time series to be embedded (where N is the count of data points).
ctDesiredDistances: $\aleph$	Count of distances to sample to generate statistics to estimate C(ε).
strFigureName: **String**	Figure name.
rgDims : $\aleph^E$ (where E is the count of embedding dimensions to	examine).
maskFigureType: $\aleph$	Type of graph to create. Must be one of the following 1: Graph showing ln(C(r)) vs ln(r) 2: Graph showing the slope of the first graph (i.e. $\frac{d \ln C(r)}{d \ln r}$ vs ln(r) (i.e. the SLOPE of figure type 1)
ctSlopeEstimationGridpoints:	Count of uniformly spaced values of ε (normalized values of r) over which to estimate the slope of the ln(C(r)) vs. lnr curve.
dMinIndexDist	Minimum distance between *indices* to be used when randomly selected points to judge distance scaling. Any pair of points selected from the time series must have a difference in indices greater than or equal to this number. The goal of such index-based filtering is to allow for less correlation between randomly selected points by filtering out points that have high correlation due to the fact that they were sampled at nearby times. (The same goal could be accomplished by downsampling the time series, but at the cost of sample counts).

Returned values

rgDim: $\aleph^E$	Specifies the dimensions for which data has been collected.

rgrgEpsilon: $\Re^{E \times R}$	Each element of this vector is the domain of C(ε) for a particular embedding dimension. For each dimension index (range 1 to E), specifies the vector of R∈ℵ normalized distance values of *ε sampled* for that dimension. Note that R will never exceed ctDesiredDistances, but could be equal to ctDesiredDistances or (for the case where certain distances were sampled multiple times) less than that value.
rgrgDEstimatedCofEpsilon : $\Re^{E \times R}$	This is the range of C(ε). For each dimension index i (range 1 to E), specifies the count of sampled distances less than or equal to each value of rgrgEpsilon{i}. In other words given 1≤i≤E and 1≤j≤R rgrgDEstimatedCofEpsilon{i}(j)=C(rgrgEpsilon{i}(j)).

Side effects

A new graph is created.

Package Uses

DetermineAndDisplayCorrelationDimensionFromTimeSeriesOnExistFig

Notes

The most common high level function for invocation.

DetermineAndDisplayCorrelationDimensionFromTimeSeriesForEmbDims

Informal description

Creates new graph and displays estimated correlation dimension (derivative of C(ε) for uniformly spaced trial embedding dimensions on that graph, as specified by minimum dimension iDimMin, maximum dimension iDimMax and dimension increment iDimIncrement.

Arguments

aTimeSeries:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
ctDesiredDistances:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
strFigureName:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
iDimMin	The minimum embedding dimension to examine.
iDimIncrement	The spacing between dimensions to examine.

iDimMax	The maximum embedding dimension to examine.
maskFigureType	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
ctSlopeEstimationGridpoints	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
dMinIndexDist	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.

Returns

rgDim	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
rgrgEpsilon	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
rgrgDEstimatedCofEpsilon	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.

Side effects

A new graph is created.

Package Uses

DetermineAndDisplayCorrelationDimensionFromTimeSeriesOnExistFig

Notes

This is a simple wrapper function for DetermineAndDisplayCorrelationDimensionFromTimeSeriesForEmbDims for the special case where the dimensions of interest are uniformly spaced.

DetermineAndDisplayCorrelationDimensionFromTimeSeriesOnExistFig

Informal description

Displays estimated correlation dimension (derivative of C(ε) for a particular embedding dimensions.

Arguments

aTimeSeries	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
ctEmbeddingDimension	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
ctDesiredDistances	Nominal dimension for the delay-reconstructed embedding of aTimeSeries.
strFigureName	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.

maskFigureType	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
ctSlopeEstimationGridpoints	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
dMinIndexDist	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.

Returns

rgEpsilon: $\Re^{\mathbb{R}}$	This is the domain of C(ε). Specifies the vector of R∈ℵ normalized distance values of *ε sampled* for that dimension. Note that R will never exceed ctDesiredDistances, but could be equal to ctDesiredDistances or (for the case where certain distances were sampled multiple times) less than that value.
rgDEstimatedCofEpsilon : $\Re^{\mathbb{R}}$	This is the range of C(ε). Specifies the count of sampled distances less than or equal to each value of rgEpsilon(i). In other words given 1≤j≤R rgDEstimatedCofEpsilon(i)=C(rgrgEpsilon(i)).

Side effects

Draws derivative of *C(ε) vs ε on existing graph.*

Package Uses

DetermineCofEpsilonFromTimeSeries
DisplayDerivativeOfCofEpsilonOnExistingFigure

Notes

Can be used to display a correlation dimension graph for a single embedding dimension.

DetermineCofEpsilonFromTimeSeries

Informal description

The highest-level function used to determine C(ε). Returns two vectors that specify successive values of ε, and C(ε) for each of these values.

Arguments

aTimeSeries	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
ctEmbeddingDimension	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.

ctDesiredDistances	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
dMinIndexDist	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.

Returns

rgEpsilon:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesOnExistFig
rgDEstimatedCofEpsilon:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesOnExistFig

Package Uses

SelectRandomDistancesFromOnTheFlyEmbeddingOfTimeSeries
DetermineCofEpsilonFromDistances

Notes

SelectRandomDistancesFromOnTheFlyEmbeddingOfTimeSeries

Informal description :

Selects ctDesiredDistances random pairs of points from the ctEmbeddingDimension-dimensional embedding of time series aTimeSeries.

Arguments

aTimeSeries:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
ctEmbeddingDimension:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesOnExistFig.
ctDesiredDistances:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
dMinDist:	Minimum distance to consider.

Package Uses

Notes

In order to prevent problems with the logarithm function elsewhere in the package, the function guards against 0-length distances by assigning an arbitrary small value (currently .1) as the distance between any points whose distance is 0. This should be replaced an alternative and more general formulation.

DetermineCofEpsilonFromDistances

Informal Description

Determines C(ε) by determining the count of datapoints (distances) in the given (unsorted) list rUnsorted that are less than or equal to different values of r. This function is simply a wrapper function for DetermineCofEpsilonFromNormalizedDistances that normalizes the given set of distances by the specified constant, and passes on the normalized distances to DetermineCofEpsilonFromNormalizedDistances.

Arguments

rUnnormalizedDistances: $\Re^n$	Array of unsorted normalized distance values.
dNormalizingCoefficient:$\Re$	Constant by which to normalize the given distances

Returns

rgEpsilon:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesOnExistFig
rgDEstimatedCofEpsilon:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesOnExistFig

Package Uses

DetermineCofEpsilonFromNormalizedDistances(rNormalizedDistances);

Notes

DetermineCofEpsilonFromNormalizedDistances

Informal Description

Determines C(ε) by determining the count of datapoints (distances) in the given (unsorted) list rUnsorted that are less than or equal to different values of ε. This is the core "workhorse" routine for determining C(ε).

Arguments

rUnsorted : $\Re^n$	Array of unsorted normalized distance values.

Returns

rgEpsilon:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesOnExistFig
rgDEstimatedCofEpsilon:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesOnExistFig

Side effects

Draws derivative of log C(ε) vs log ε on existing graph.

Package Uses

None

Notes

DisplayDerivativeOfCofEpsilonOnExistingFigure

Informal description

Displays the estimated derivative of log C(ε) vs. logε on a preexisting graph.

Arguments

rgEpsilon: $\Re^m$	Sorted (in ascending order) unique vector of values of ε
rgDEstimatedCofEpsilon: $\Re^m$	Vector of C(ε) where ε is as specified by rgEpsilon. rgDEstimatedCofEpsilon[i] = | { r ∈ rUnsorted s.t. r < rgEpsilon[i] } |

ctSlopeEstimationGridpoints:ℵ The count of points at which to estimate the slope of the log(C(ε)) vs log(ε) graph.

Side effects

Draws derivative of log C(ε) vs log ε on existing graph.

Package Uses

None

Notes

DetermineAndDisplayCorrelationDimFromSynthStructOnExistFig

Informal description

Displays the estimated derivative of log C(ε) vs. logε from a synthetic structure of dimension ctStructureDim on a preexisting graph.

Arguments

ctStructureDim: $\Re^m$	The intrinsic dimension of the structure which to create.
ctEmbeddingDimension: $\Re^m$	The dimension in which to embed the synthetic structure.
ctDesiredDistances:ℵ	See DetermineAndDisplayCorrDimensionFromTimeSeriesForRgEmbDims.
strFigureName:ℵ	See DetermineAndDisplayCorrDimensionFromTimeSeriesForRgEmbDims.
maskFigureType:ℵ	See DetermineAndDisplayCorrDimensionFromTimeSeriesForRgEmbDims.
ctSlopeEstimationGridpoints:ℵ	The number of gridpoints used in slope termination for estimating the correlation dimension..

Side effects

Draws derivative of log C(ε) vs log ε on existing graph.

Package Uses

SelectRandomDistancesFromNDimensionalStructureInSampleSpace
DetermineCofEpsilonFromDistances
DisplayDerivativeOfCofEpsilonOnExistingFigure

Notes

DetermineAndDisplayCorrelationDimFromSynthStructCtSamples

Informal description

Displays the estimated derivative of log C(ε) vs. logε from a synthetic structure of dimension ctStructureDim on a new graph, but for different sample counts

Arguments

ctStructureDim: $\Re^m$	The intrinsic dimension of the structure which to create.
ctDesiredDistancesMin:ℵ	Minimum count of desired distances to sample when determining the correlation dimension.

ctDesiredDistancesIncr:ℵ	Increment count of desired distances to sample when determining the correlation dimension.
ctDesiredDistancesMax:ℵ	Maximum count of desired distances to sample when determining the correlation dimension.
strFigureName:ℵ	See DetermineAndDisplayCorrDimensionFromTimeSeriesForRgEmbDims.
ctEmbeddingDimension: $\Re^m$	The dimension in which to embed the synthetic structure.
maskFigureType:ℵ	See DetermineAndDisplayCorrDimensionFromTimeSeriesForRgEmbDims.
ctSlopeEstimationGridpoints:ℵ	The number of gridpoints used in slope termination for estimating the correlation dimension..

Side effects

Draws derivative of log C(ε) vs log ε on new graph.

Package Uses

DetermineAndDisplayCorrelationDimFromSynthStructOnExistFig

Notes

SelectRandomDistancesFromNDimensionalStructureInSampleSpace

Informal description

Randomly selects distances from a synthetic structure of dimension within a.

Arguments

ctStructureDim: $\Re^m$	See DetermineAndDisplayCorrelationDimFromSynthStructOnExistFig.
ctEmbeddingDimension: $\Re^m$	See DetermineAndDisplayCorrelationDimFromSynthStructOnExistFig.
ctDesiredDistances:ℵ	See DetermineAndDisplayCorrDimensionFromTimeSeriesForRgEmbDims.

Returns

Returns an array: $\Re^{ctDesiredDistances}$ of random distances.

Side effects

Package Uses

CreateDDimensionalVectorInNDimensionalSubspace

Notes

CreateDDimensionalVectorInNDimensionalSubspace

Informal description

Displays the estimated derivative of log C(ε) vs. logε on a preexisting graph.

Arguments

ctStateSpaceDim: ℵ	Count of dimensions to populate with uniform values drawn from the interval [0 1]. All other vector components will have value 0
ctStructureDim: ℵ	Total length of vector.

Side effects

None.

Package Uses

None.

Notes

DetermineAndDisplayCorrelationDimensionFromTimeSeries

Informal description

Displays the estimated derivative of log C(ε) vs. logε on a new graph for a particular dimension.

Arguments

aTimeSeries:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
ctEmbeddingDimension:	"Trial" embedding dimension in which to display the empirical estimate of the correlation dimension.
ctDesiredDistances	Count of distance samples to take when estimating the correlation dimension.
strFigureName:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
maskFigureType	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
ctSlopeEstimationGridpoints	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
dMinIndexDist	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.

Return Values

rgEpsilon	See definition in *DetermineAndDisplayCorrelationDimensionFromTimeSeriesOnExistFig*
rgDEstimatedCofEpsilon	See definition in *DetermineAndDisplayCorrelationDimensionFromTimeSeriesOnExistFig*

Side effects

Creates a new figure window, displaying the estimated correlation dimension for different values of the normalized distance ε.

Package Uses

None.

Notes

6.2 Testing and Support Functions

DetermineAndDisplayCorrelationDimensionFromSynthStructInRgEmbDims

Informal description

Creates a synthetic structure, assesses its dimensionality at a specified set of trial embedding dimension, and displays the result on a new figure.

Arguments

ctStructureDim:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
ctDesiredDistances:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
strFigureName :	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
rgDims:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
maskFigureType:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
ctSlopeEstimationGridpoints:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.

Returned values

rgDim:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
rgrgEpsilon:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
rgrgDEstimatedCofEpsilon:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.

Side effects

Displays a graph on a new structure.

Package Uses

DetermineAndDisplayCorrelationDimFromSynthStructOnExistFig

Notes

DetermineAndDisplayCorrelationDimensionFromSynthStructInEmbDims

Informal description

Creates a synthetic structure, assesses its dimensionality at a specified set of trial embedding dimension, and displays the result on a new figure.

Arguments

ctStructureDim: See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.

ctDesiredDistances: See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.

strFigureName : See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.

iDimMin : See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForEmbDims.

iDimIncrement : See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForEmbDims.

iDimMax: See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForEmbDims.

maskFigureType See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.

ctSlopeEstimationGridpoints: See definition in

Returned values

rgDim: See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.

rgrgEpsilon: See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.

rgrgDEstimatedCofEpsilon: See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.

Package Uses

DetermineAndDisplayCorrelationDimensionFromSynthStructInRgEmbDims

Side effects

Displays a graph on a new structure.

Notes

EstimateMaxEmbeddingDistanceFromTimeSeries

Informal description

Derives an estimated (approximate) maximum of the distances in the ctEmbeddingDimension-dimensional embedding of time series aTimeSeries. This approximation is contracted through deriving the maximum of a bootstrapped sample of size ctSamplePairs within the embedding of aTimeSeries. In particular, the function derives an approximation to the maximum by deriving the maximum of the distances between ctSamplePoints pairs of points each element of which is drawn randomly from the ctEmbeddingDimension embedding of aTimeSeries.

Arguments

aTimeSeries: See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.

ctSamplePairs:	Set of randomly selected sample pairs between which to judge distances
ctEmbeddingDimension:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesOnExistFig.

Package Uses

None

Returned values

dEstimatedMax:	Estimated maximum distance in the data set, as judged from ctSamplePairs sample pairs drawn from aTimeSeries.

Side effects

None.

Notes

Not currently used by other functions.

Display3DEmbeddingSpace

Informal description

Displays a scatterplot of embedding of the given time series in 3D.

Arguments

aTimeSeries:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
strFigureName:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
dPointSize: ℵ	Size of the datapoints to display.

Returned values:

None.

Side effects

Displays a graph on a new structure.

Package Uses

None.

Notes

DLaggedCovariance

Informal description

Determines the covariance between time series aTimeSeries and itself, lagged by index count iLag. Using this function with iLag=0 will cause it to return of aTimeSeries.

Arguments

aTimeSeries:	See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.
dLag: ℵ	Lag with which to determine the coveriance.

DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.

Returned values

`The specified covariance (ℜ).`

Package Uses

None.

Side effects

None.

Notes

DeriveExhaustiveNDimensionalEmbedding

Informal description

Determines the covariance between time series aTimeSeries and itself, lagged by index count iLag. Using this function with iLag=0 will cause it to return of aTimeSeries.

Arguments

`aTimeSeries:`	`See definition in DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.`
`dLag: ℵ`	`Lag with which to determine the coveriance.`

`DetermineAndDisplayCorrelationDimensionFromTimeSeriesForRgEmbDims.`

Returned values

`The specified covariance (ℜ).`

Package Uses

None.

Side effects

None.

Notes

7 Acknowledgements

The author acknowledges his gratitude to Mark Smith for discussions regarding dynamical systems.

8 Bibliography

1. Osgood, N., *Representing Heterogeneity in Complex Feedback System Modeling: Computational Resource and Error Scaling*, in *Presented at the 22nd International Conference of the System Dynamics Society*. 2004, International System Dynamics Society: Oxford, UK.
2. XJ Technologies, *AnyLogic*. 2005, XJ Technologies: St. Petersburg, Russia.
3. Osgood, N., *Motivations for the use of ABM and ABM Frameworks: One Practitioner's Perspective*, in *International Conference on System Dynamics 2005*. 2005: Boston.
4. Rahmandad, H. and J. Sterman. *Heterogeneity and Network Structure in the Dynamics of Contagion: Comparing Agent-based and Differential Equation Models*. in *22nd Annual International Conference on System Dynamics* 2004. Oxford: System Dynamics Society.

5. Zaric, G.S., *Random vs. nonrandom mixing in network epidemic models.* Health care management science, 2002. **5**(2): p. 147-55.

6. Koopman, J., S. Chick, and G. Jacquez, *Analyzing sensitivity to model form assumptions of infection transmission.* Annals of Epidemiology, 2000. **10**(7): p. 472.

7. Koopman, J., *Modeling Infection Transmission.* Annual Review of Public Health, 2004. **25**(1): p. 303-326.

8. Jacquez, G.M., et al. *Complex systems analysis using space-time information systems and model transition sensitivity analysis*. in *6th International Symposium on Spatial Accuracy Assessment in Natural Resources and Environmental Sciences*. 2004. Portland, Maine: International Environmetrics Society, St. Lucia, Queensland, Australia.

9. Osgood, N., *Multi-Framework Simulation Modeling for Decision Making*. 2005.

10. Kantz, H. and T. Schreiber, *Nonlinear Time Series Analysis*. Second Edition ed. 2004, New York: Cambridge University Press. 369.

11. Takens, F., *On the numerical determination of the dimension of an attractor*, in *Lecture Notes in Mathematics 898*. 1981, Springer-Verlag. p. 366-381.

12. Packard, N.H., et al., *Geometry from a Time Series.* Physical Review Letters, 1980. **45**(9): p. 712-716.

13. Grassberger, P. and I. Procaccia, *Measuring the strangeness of strange attractors.* Physica A, 1983. **9**(1-2): p. 189.

14. Grassberger, P. and I. Procaccia, *Characterization of Strange Attractors.* Physics Review Letters, 1983. **50**: p. 346.

15. Camastra, F., *Estimating the intrinsic dimension of data with a fractal-based method.* IEEE transactions on Pattern Analysis and Machine Intelligence, 2002. **24**(10): p. 1404.

16. Smith, L.A., *Intrinsic Limits on Dimension Calculations.* Physics Letters, 1988. **A133**(6): p. 283-288.

17. Eckmann, J.P. and D. Ruelle, *Fundamental Limitations for Estimating Dimensions and Lyapunov Exponents in Dynamical Systems.* Physica D, 1992. **56**: p. 185-187.

18. Lorenz, E.N., *Deterministic nonperiodic flow.* Journal of Atmospheric Science, 1963. **20**: p. 130.

19. Gilmer Jr, J.B. and F. Sullivan. *Dimensionality analysis of a simulation outcome space*. in *2001 Winter Simulation Conference*. 2001: IEEE Computer Society