Chapter 1

Perturbation Analysis: A Complex Systems Pattern

Nicholas Geard, Kai Willadsen and Janet Wiles
ARC Centre for Complex Systems,
School of Information Technology and Electrical Engineering,
The University of Queensland, QLD 4072 Australia.
E-mail: {nic,kaiw,janetw}@itee.uq.edu.au

Patterns are a tool that enables the collective knowledge of a particular community to be recorded and transmitted in an efficient manner. Initially developed in the field of architecture and later developed by software engineers [6], they have now been adopted by the complex systems modelling community [15]. It can be argued that, while most complex systems models are idiosyncratic and highly specific to the task for which they are constructed, certain tools and methodologies may be abstracted to a level at which they are more generally applicable. This paper presents one such pattern, Perturbation Analysis, which describes the underlying framework used by several analytical and visualisation tools to quantify and explore the stability of dynamic systems. The format of this paper follows the outline specified in [15].

Pattern name Perturbation Analysis
Classification Dynamics, State Space
Intent The Perturbation Analysis pattern provides a quantifiable measure of the stability of a dynamic system.
Also known as None

Motivation

A complex dynamic system is one consisting of multiple elements, where the future state of the system is determined by a function f of its current state,

$$\mathbf{s}(t+1) = f(\mathbf{s}(t))$$

where $\mathbf{s}(t)$ is the state of the system at time t.

The typical feature of interest of complex dynamic systems is their asymptotic behaviour as $t \rightarrow \infty$. The set of states towards which a system converges under these conditions is known as an *attractor*. Attractors may be fixed points, limit cycles, or non-repeating 'chaotic' attractors. Systems may contain single or multiple attractors. The set of initial states of a system that converge to a given attractor forms the *basin of attraction* of that attractor.

Complex dynamic systems are widespread: genetic networks, economies and ecosystems are all examples. One of the emergent features of these types of distributed systems is their robustness or stability. Systems in the real world operate in noisy environments and are subject to perturbations from a wide variety of internal and external sources. In many cases, despite short term fluctuations, the long term behaviour of these systems is remarkably stable.

When modelling such systems, it is useful to be able to quantify this level of stability. For example, in a genetic regulatory system, where basins of attraction have been equated to cell behaviours [10], the stability of a system may reflect the phenomena of cell differentiation during development. Early in the developmental process, cells are sensitive to signals from their environment: transplantation experiments have demonstrated how embryonic cells can adopt the fate of their new neighbours rather than their original fate. As development progresses, the stability of cell types increases, and almost all fully differentiated cells will retain their original fate when transplanted [17]. The differentiation process itself is robust to fluctuations in external factors, such as temperature variation and nutrient levels, as well as internal factors, such as the stochastic nature of many genetic and cellular processes [11]

The Perturbation Analysis pattern provides a general framework for measuring the effect of changes to a system's current state on its long-term behaviour. These measurements may then be used as the basis for calculating more specific quantities, such as the rate of convergence or divergence of two nearby trajectories, or the probability of a perturbation causing a system to switch between different attractors.

Applicability

The Perturbation Analysis pattern requires a dynamic system, consisting of:

- a finite set of elements, each of which may take a discrete or continuous value; and
- a deterministic updating function.

The Perturbation Analysis pattern is useful in the following situations:

(1) A dynamic system is subject to some intrinsic or extrinsic perturbation, and it is desirable to stochastically or systematically explore and quantify the effects of these perturbations.
(2) A dynamic system can settle down into one of several possible behaviours and it is desirable to know either the likelihood of a system reaching a specific stable behaviour, or the probability of a system switching from one stable behaviour to another.
(3) A dynamic system is being used for prediction and it is desirable to know how far into the future its behaviour can be confidently predicted if there is some uncertainty as to its initial state.

Structure

The relationships between the classes involved in the Perturbation Analysis pattern are detailed in Figure 1.1.

Participants

State stores a state of the system $\mathbf{s}$, represented as a vector the values of each of the n elements,

$$\mathbf{s} = (s_0, \ldots, s_{n-1})$$

System applies an update function f to update the values of each element of a state,

$$\mathbf{s}(t+1) = f(\mathbf{s}(t))$$

Perturber applies a perturbation function p to create a new state from an old state in a systematic fashion,

$$\mathbf{s}' = p(\mathbf{s})$$

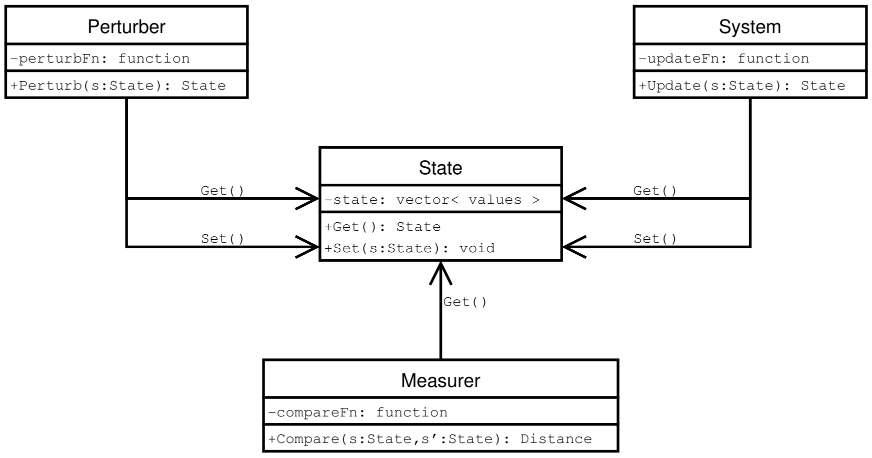


Fig. 1.1 A class diagram describing the types of objects involved in the Perturbation Analysis pattern and the relationships that exist between them. Each object lists the private data variables it contains (indicated by a minus), and the public functions it provides (indicated by a plus), together with their arguments and return values.

Measurer quantifies the distance d between two states according to some metric m,

$$d = m(\mathbf{s}, \mathbf{s}')$$

Concrete examples of the update and perturbation functions, and of the distance metric, are provided below, in the Implementation Section.

Collaborations

A dynamic view of the interactions between objects in the Perturbation Analysis pattern is shown in Figure 1.2.

(1) Note that two States are maintained at all times: one ($\mathbf{s}$) corresponding to the original system trajectory and another ($\mathbf{s}'$) to the perturbed trajectory.
(2) Perturber sets the value of the perturbed State according to the application of the Perturb function to the original State.
(3) Measurer uses the Distance metric to calculate the distance between the original and perturbed trajectories.
(4) System uses the Update function to advance each of the states by one iteration (or time step).

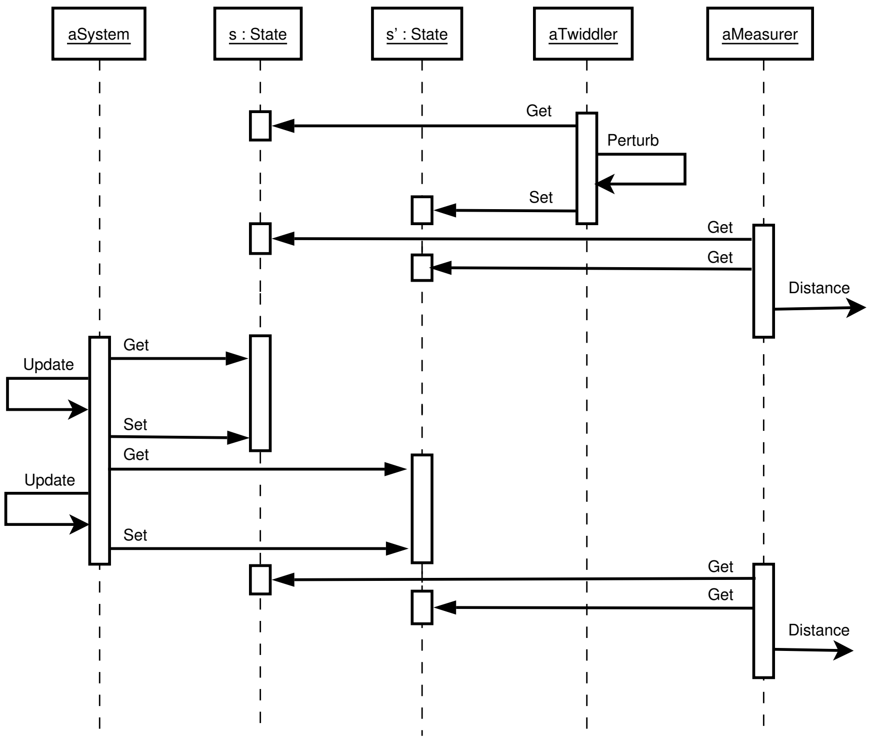


Fig. 1.2 A sequence diagram of the interactions between objects in the Perturbation Analysis pattern. Time runs vertically from top to bottom. The activation bars in the lifeline of each object indicate when that object is active in the interaction. Horizontal arrows indicate interactions – in this case function calls.

Consequences

The Perturbation Analysis pattern has the following benefits and limitations:

(1) The pattern facilitates perturbation of a system and collation of distance measurements which can then be analysed using other methods. Two examples of the type of context in which the Perturbation Analysis pattern can be applied are provided below in the Sample Code Section.
(2) The pattern allows for a range of perturbation functions, distance metrics and system updating functions, each of which can be varied independently. Examples of these functions and metrics are provided below in the Implementation Section.
(3) Because the pattern only specifies a single iteration of the perturb

and measure cycle, it supports the investigation of both annealed and quenched systems. In a quenched system, the structure of the system and the update function are static through time: measurements of a quenched system are specific to that particular instance of the system. In an annealed system, the basic parameters of the system (level of connectivity and type of updating function) are static, but the specific pattern of connectivity and set of updating functions are generated anew at each time step: measurements of an annealed system reflect basic properties of an entire class of systems.

(4) One limitation of the pattern as described here is that it requires a deterministic system updating function. While there is no reason that the pattern could not be applied to a stochastic system, doing so raises several issues that have not been addressed here relating to the structure of state spaces and the nature of attractors (see, e.g., [8]).

Implementation

The Perturbation Analysis pattern is generally applied as an iterative procedure. That is, a large number of perturbations and measurements are carried out in order to provide an estimation of the stability of a particular system or class of systems. A single iteration of perturbation and measurement may be described (in pseudocode) as follows:

```
# Set the initial state of the system.
State s = initialState

# Perturb the current state.
State s' = Perturber.Perturb (s)

# Measure the distance between the original and perturbed states.
startDist = Measurer.Distance (s, s')

# Update both the original and perturbed states.
s = System.Update (s)
s' = System.Update (s')

# Measure the distance between the updated states.
endDist = Measurer.Distance (s, s')
```

The main variables in this procedure are the nature of the update and perturbation functions and the distance metric. Each of these aspects may be varied independently.

(1) *Defining an update function.* The update function is defined by the dynamic system to which the Perturbation Analysis pattern is being applied. An example of an updating function in a discrete dynamic system is provided by Kauffman's Random Boolean Network model [10]. In this model, the value of a state element, σ_n, at time $t+1$ is some random Boolean function, f_n, of its K inputs at time t,

$$f_n(t+1) = f_n(\sigma_{n_1}(t), \dots, \sigma_{n_K}(t))$$

An example of a continuous update function is the sigmoid function used in many neural network applications,

$$f(x) = \frac{1}{1+e^{-x}}$$

where x is a weighted sum of the inputs to a particular element.

(2) *Defining a perturbation function.* Perturbation functions on discrete states can be either systematic or stochastic. A systematic perturbation function varies state elements systematically (e.g., by incrementing or decrementing an integer value, or by negating a Boolean value). A stochastic perturbation function varies state elements without regard for the sequential nature of the element values (e.g., randomly assigning a new integer value within the allowable range). A single application of either type of perturbation function will involve the alteration of one or more elements. The properties of the perturbation function are therefore:

- the number of elements being varied; and
- the mechanism (i.e., systematic or stochastic) used in their variation.

Several possibilities exist for perturbing continuous state element values. Unlike the discrete case, it is not possible to systematically explore the set of *all* possible perturbations. Therefore perturbations are generally applied in a stochastic fashion, by the addition of noise generated according to some distribution (e.g., Uniform or Gaussian) to some or all of the elements. The properties of the perturbation function that can be modified are:

- the number of elements modified by the addition of noise; and
- the parameters of the distribution used to generate the noise, for example, the mean and standard deviation of a Gaussian distribution.

Another possibility for perturbing continuous state elements is to define a discrete-valued structure embedded within the continuous state space and then systematically perturb the system within the bounds defined by this structure. For example, consider a system with three elements, in which the values of each element are constrained to the range $[0, 1]$. It is possible to define a three-dimensional cube within this space and constrain the initial states and perturbations to the vertices of the cube. If greater resolution is desired, the cube may be subdivided to introduce the midpoints of the edges and the centre of the cube. This method enables a continuous space to be explored and perturbed in a systematic fashion.

(3) *Defining a distance function.* In the case of discrete states, a distance function applies some transformation to the set of distances between individual elements of the state. The most common transformation performed in discrete systems is summation (e.g., Hamming distance), though other transformations such as the average or the sum of squares may be used.

When the values of the state elements are continuous, the standard distance metric is the Euclidean distance between the original and perturbed states. For a system with N elements, the Euclidean distance m between states $\mathbf{s}$ and $\mathbf{s}'$ is given by,

$$m(\mathbf{s}, \mathbf{s}') = \left[\sum_{i=1}^{N} (\mathbf{s}'_i - \mathbf{s}_i)^2 \right]^{\frac{1}{2}}$$

where $\mathbf{s}_i$ is the value of the ith element of state $\mathbf{s}$.

(4) *Alternative distance measures.* Some methods for measuring the effects of perturbations do not require the initial distance measurement indicated above. An example method is the standard basin of attraction stability measurement commonly used in Random Boolean Network models [12; 2] – in this analysis the distance comparison is based solely on the final basins of attraction of the perturbed and unperturbed states.

Sample code

Boolean network attractor stability

A common application of the Perturbation Analysis pattern is to estimate the stability of an attractor in a Boolean network through either stochastic or systematic perturbation of attractor states. The typical unit of measurement in this usage case is whether or not the perturbed state reaches the same basin of attraction as the unperturbed state. Repeated trials are used to provide an estimation of the stability of a given basin of attraction, where stability is defined as the probability that a perturbation to a state

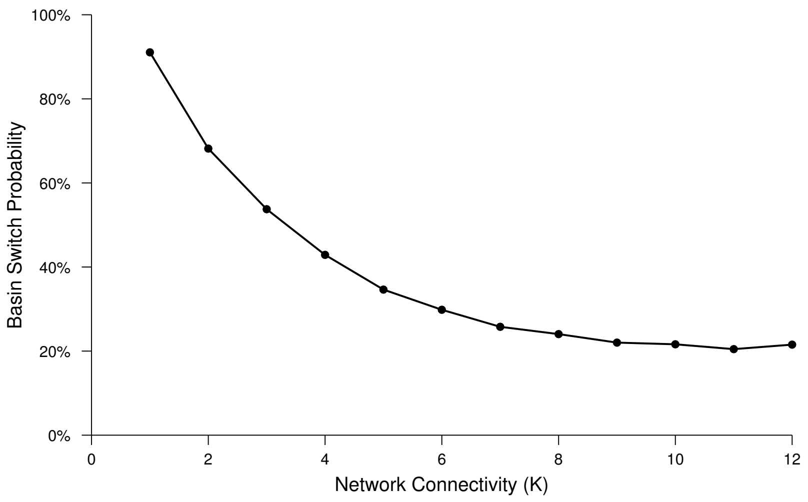


Fig. 1.3 Variation of attractor stability with increasing degree of connectivity in an $N = 12$ random Boolean network.

does not change the basin of attraction.

One standard approach to obtaining such a stability measurement is to look only at the states in the attractor [8; 12; 2]. In this situation, the resulting measurement is the probability that the perturbation of an attractor state will move the system to a different basin of attraction.

(1) Choose a state $\mathbf{s}$ in attractor a.
(2) Perturb $\mathbf{s}$ to obtain $\mathbf{s}'$. This step is generally performed by flipping n elements of the Boolean state, where n is a small integer (frequently one).
(3) Iterate the trajectory starting at $\mathbf{s}'$ until it reaches an attractor a'.
(4) Store the value $a = a'$.
(5) Repeat steps 1 to 4 some number of times and calculate the average of the values stored in step 4.

An interesting characteristic of Boolean networks is the change in their behaviour as the degree of connectivity of the network (K) varies.

By repeated application of the above procedure, an approximation of the stability of the attractor states can be obtained for a range of connectivity values. Figure 1.3 shows the results of measuring stability in the above manner on a Random Boolean Network model with $N = 12$, by iterating through N perturbations of all 2^N system states and recording the probability of the target attractors of the original and perturbed states being different. This observed decrease in system stability is consistent with general expectations of the behaviour of the Random Boolean Network model.

Lyapunov characteristic exponents

One purpose for which the Perturbation Analysis pattern may be applied is estimating the largest Lyapunov exponent in order to determine the stability of an attractor. The Lyapunov exponents of a system measure the exponential rate of convergence or divergence of two nearby trajectories. If the largest Lyapunov is negative, the attractor is stable. If the largest Lyapunov is positive, the attractor is chaotic, and the magnitude of the exponent gives an indication of the time scale on which the future behaviour of the system becomes unpredictable.

The Lyapunov exponent λ is given by,

$$\lambda = \lim_{t \to \infty, \delta x_0 \to 0} \frac{1}{t} log \frac{|\delta x_t|}{|\delta x_0|}$$

where δx_t is the separation of the original and perturbed trajectories at time t. While methods do exist for determining the Lyapunov exponent directly from the equations describing a system's dynamics, it is also possible to approximate the value from a series of data points. The procedure for estimating the largest Lyapunov exponent is as follows [14]:

(1) Choose an initial system state.
(2) Iterate the state $\mathbf{s}$ until it is located on an attractor.
(3) Perturb $\mathbf{s}$ to obtain $\mathbf{s}'$. This step is generally performed by adding a small amount of Gaussian noise (mean 0, standard deviation 1×10^{-9}) to each of the state elements.
(4) Calculate the Euclidean distance d_0 between $\mathbf{s}$ and $\mathbf{s}'$.
(5) Iterate both trajectories.
(6) Calculate the new Euclidean distance d_1.
(7) Calculate and store the value $log|\frac{d_1}{d_0}|$.
(8) Perturb $\mathbf{s}$ to obtain $\mathbf{s}'$ such that distance between them is d_0 in the direction of d_1. This step can be carried out by adjusting each element i of state $\mathbf{s}'$ such that,

$$\mathbf{s}'_i = \mathbf{s}_i + \frac{d_0(\mathbf{s}'_i - \mathbf{s}_i)}{d_1}$$

(9) Repeat steps 5 to 8 some number of times and calculate the average of the values stored in step 7.

The number of iterations required to reach an attractor in step 2 and the number of iterations of steps 5 to 8 required for the value of λ to converge may vary. Similarly, it can be useful to repeat the calculation process using different initial states (step 1) and different initial perturbations (step 3). It is important to note that if a system contains more than one attractor, then the value of λ will be specific to the particular basin of attraction that contains the initial state.

One way to analyse the behaviour of a dynamical system is to explore the behaviour of a family of parameterised functions. For example a family of linear systems may be described by the function $f_m(x) = mx$ where m is varied over the real numbers. It is then possible to observe how the dynamics of the system change as the function is changed. The same technique may be applied to investigate the behaviour of more complex systems, such as neural networks, by the inclusion of a gain parameter g that scales the net input into the update function f,

$$f_g(x) = \frac{1}{1 + e^{-gx}}$$

As g affects the slope of the sigmoid function, modifying g from very small to very large results in a sweep from the linear range, through the nonlinear range to the Boolean range when the function is saturated.

By calculating the value of the Lyapunov exponent (using the same initial state each time) for each value of g, the range of dynamic behaviours of a particular system can be visualised. Figure 1.4 shows how perturbation analysis may be used to visualise the dynamics of a recurrent neural network [5]. The network used in this example consisted of 20 fully-connected nodes, with weights drawn from a Gaussian distribution with mean 0 and standard deviation 1. The gain parameter g was varied from 0.2 and 40 with increments of 0.2. An initial system state I was generated by setting the activation of each node to a value in the range [0, 1]. For each value of g the system was initialised to I and the procedure described above was used to estimate the Lyapunov exponent (Figure 1.4, top). In addition, the average activation of the network was recorded for each iteration of the calculation, providing an alternative visualisation of network dynamics (Figure 1.4, bottom). These two complementary views provide a comprehensive picture of the dyanmics of a system across a range of weight scales, revealing such features as bifurcations, fixed point and cyclic attractors, and chaotic behaviour.

Known uses

The concept of perturbation analysis as an exploratory tool was first formalised in the realm of Discrete Event Dynamic Systems, where it was developed to estimate the gradient of performance measures with respect to variation in system control parameters (see [9] for a history and overview of perturbation analysis in this context).

Within the field of complex systems, perturbation analysis has been used on an *ad hoc* basis by numerous researchers as a means of exploring the stability of genetic regulatory systems (e.g., [12]). Perturbation analysis has also been employed in a more principled fashion, to generate theoretical results about system stability: Derrida's annealed approximation method [4] illustrates the use of the Perturbation Analysis pattern on an annealed version of the Random Boolean Network model. This analytic tool uses an annealed random Boolean updating function, a stochastic perturbation process involving all N state elements and a state distance metric based on the normalised overlap of the states' values. The annealed approximation method was used to show that $K = 2$ connectivity in the Random Boolean Network model described a phase transition between the ordered and chaotic behaviour of the system. The annealed approximation method has since been used in different situations to identify phase transitions in the behaviour of networks of multi-state automata [13], and Boolean networks with scale-free topologies [2].

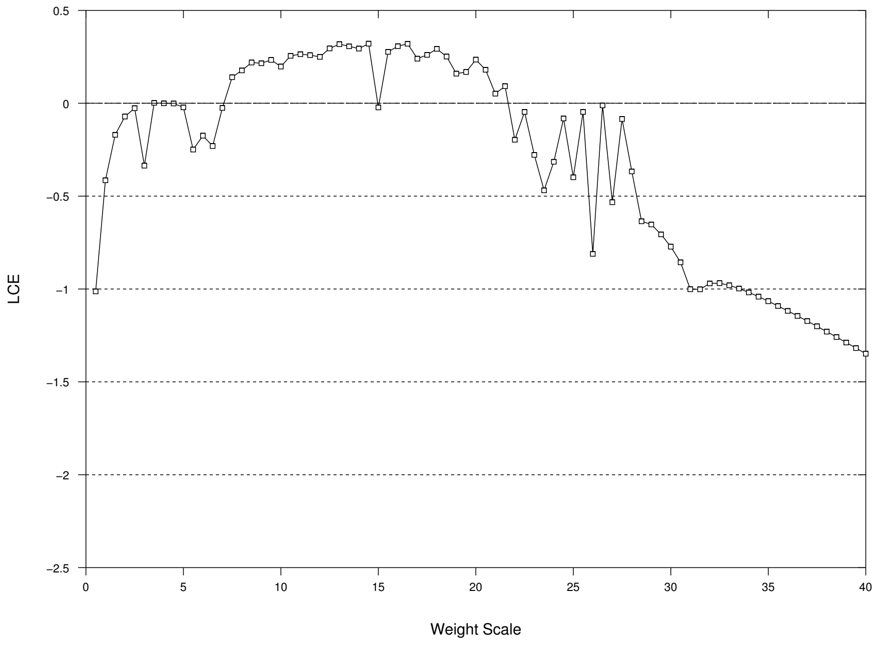
Lyapunov exponents have been used by mathematicians as an indicator of chaotic systems for some time. During the 1980s, several approaches were developed to allow the Lyapunov exponent to be determined from time series data [16], allowing the recognition of chaos in systems whose generating equations were unknown. Subsequent studies introduced the use of neural networks as general models of dynamic systems, typically for econometric and financial time series prediction tasks (e.g., [3]). More recently, simulations of high dimensional neural networks and systematic measurement of Lyapunov exponents has been used to investigate routes to chaos in high dimensional nonlinear systems [1]. Finally, the techniques described here have been extended and used to develop intuitions about the formation and stability of attractors in network models of gene regulation [7].

Summary

This paper has used the formal framework of patterns to describe a standard technique for analysing the stability of complex dynamical systems. The Perturbation Analysis pattern can be applied to a variety of discrete and continuous systems, as demonstrated by the random Boolean network and neural network examples detailed above. This form of stability analysis allows the effects of intrinsic and extrinsic perturbations on the dynamics of a system to be quantified. This paper also serves as an example of how the software engineering concept of patterns can be used to formalise modelling techniques and strategies for effective communication within a research community.

Acknowledgements

This pattern was developed during a Patterns Workshop held by the ARC Centre for Complex Systems on 6-7 July 2005.



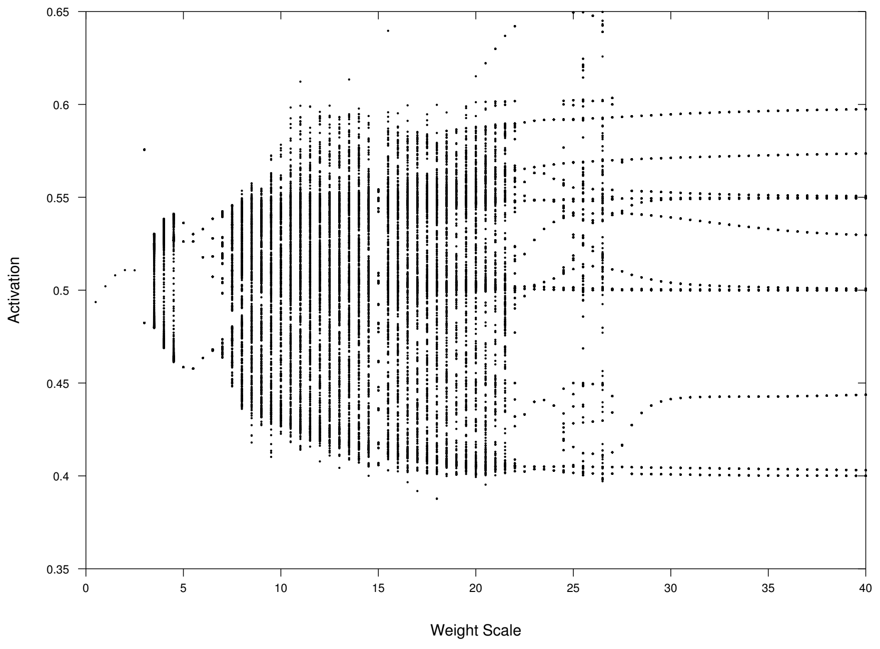


Fig. 1.4 Lyapunov exponent (top) and activation diagram (bottom) for a fully connected 20 node network as g is scaled from 0 to 40. Note the correlation between fixed point and cyclic attractors, indicated by single or multiple discrete points on the bottom chart, with negative Lyapunov values. In contrast chaotic attractors, with positive Lyapunov, values appear as as 'smears' of points.

Bibliography

D. J. Albers, J. C. Sprott, and W. D. Dechert. Routes to chaos in neural networks with random weights. *International Journal of Bifurcation and Chaos*, 8(7):1463–1478, 1998.

M. Aldana. Boolean dynamics of networks with scale-free topology. *Physica D*, 185:45–66, 2003.

W. D. Dechert and R. Gencay. Lyapunov exponents as a nonparametric diagnostic for stability analysis. *Journal of Applied Econometrics*, 7:S41–S60, 1992.

B. Derrida and Y. Pomeau. Random networks of automata: A simple annealed approximation. *Europhysics Letters*, 1:45–49, 1986.

J. L. Elman". Finding structure in time. *Cognitive Science*, 14:179–211, 1990.

E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns*. Addison Wesley, New York, NY, 1995.

N. Geard. The control of complexity: Modelling the role of noncoding RNA in gene regulation. PhD Thesis – in preparation, 2005.

I. Harvey and T. Bossamaier. Time out of joint: Attractors in asynchronous random Boolean networks. In P. Husbands and I. Harvey, editors, *Fourth European Conference on Artificial Life*, Cambridge, MA, 1997. The MIT Press/Bradford Books.

Y.-C. Ho. Perturbation analysis: Concepts and algorithms. In J. J. Swain, D. Goldsman, R. C. Crain, and J. R. Wilson, editors, *Proceedings of the 1992 Winter Simulation Conference*, Piscataway, NJ, 1992. The IEEE Press.

S. A. Kauffman. *The Origins of Order: Self-Organization and Selection in Evolution*. Oxford University Press, Oxford, UK, 1993.

H. Kitano. Biological robustness. *Nature Reviews Genetics*, 5:826–837, 2004.

T. Reil. Dynamics of gene expression in an artificial genome - implications for biological and artificial ontogeny. In D. Floreano, J.-D. Nicoud, and F. Mondada, editors, *Advances in Artificial Life – ECAL 1999: 5th European Conference*, Berlin, 1999. Springer-Verlag.

R. V. Solé, B. Luque, and S. Kauffman. Phase transitions in random networks with multiple states. Working Paper 00-02-011, Santa Fe Institute, 2000.

J. C. Sprott. Numerical calculation of largest Lyapunov exponent.

http://sprott.physics.wisc.edu/chaos/lyapexp.htm, 2004.

J. Wiles and J. Watson. Patterns in complex system modeling. In M. Gallagher, J. Hogan, and F. Maire, editors, *Intelligent Data Engineering and Automated Learning – IDEAL 2005: 6th International Conference*, Berlin, 2005. Springer-Verlag.

A. Wolf, J. B. Swift, H. L. Swinney, and J. A. Vastano. Determining Lyapunov exponents from a time series. *Physica D*, 16:285–317, 1985.

L. Wolpert. *The Principles of Development.* Oxford University Press, Oxford, UK, 1998.