Time-Series Based Prediction of Dynamical Systems and Complex Networks

Celso Grebogi

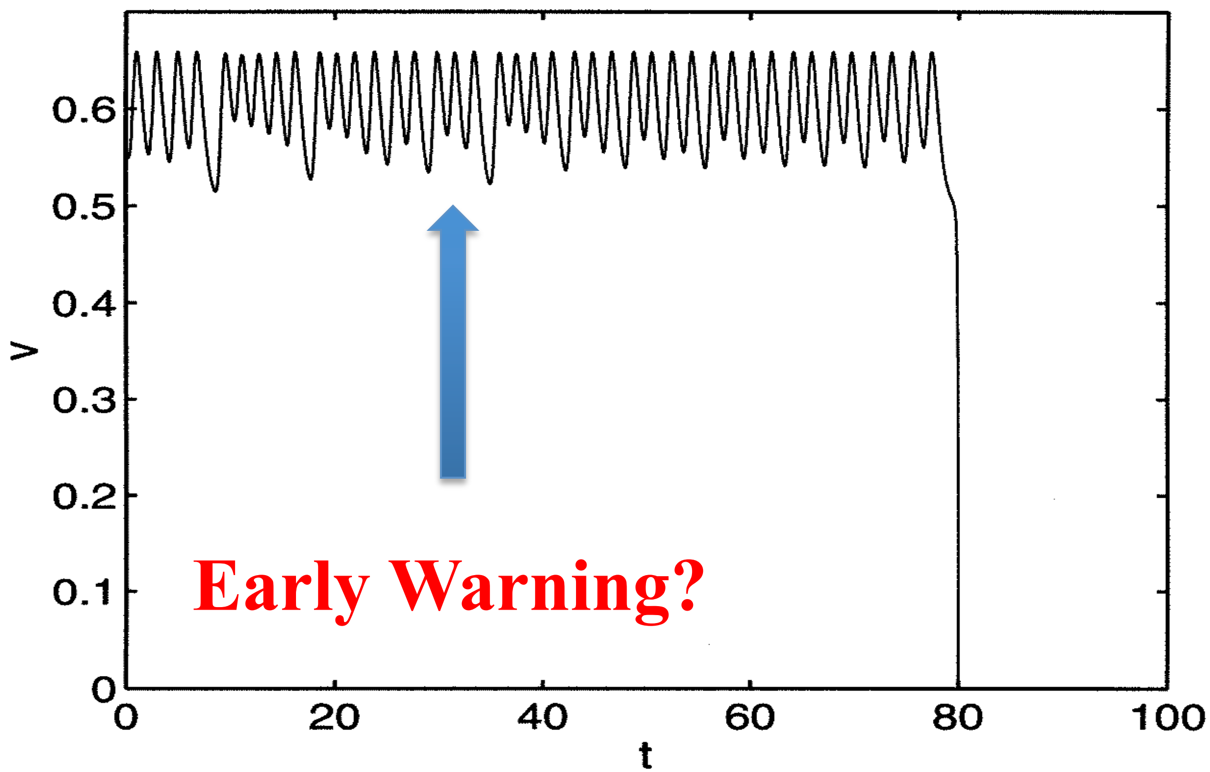
University of Aberdeen, UK

http://www.abdn.ac.uk/icsmb/people/details/grebogi

Collaborators:

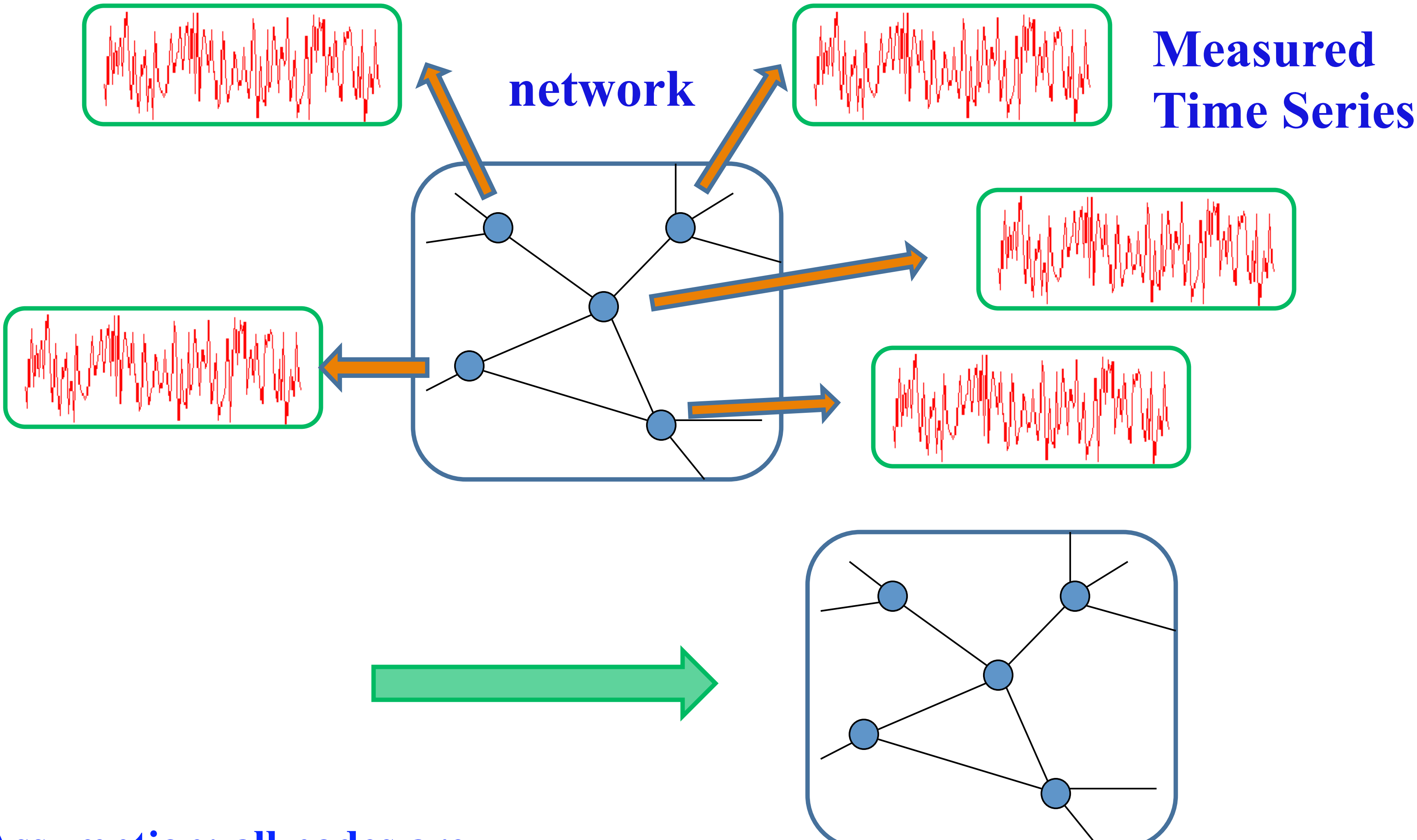
Dr. Wenxu Wang, ASU
Mr. Ryan Yang, ASU
Mr. Riqi Su, ASU
Prof. Ying-Cheng Lai, ASU & Univ. of Aberdeen
Dr. Vassilios Kovanis, AFRL, Wright-Patterson AFB

Problem 1: Can catastrophic events in dynamical systems be predicted in advance?



A related problem: Can future behaviors of time-varying dynamical systems be forecasted?

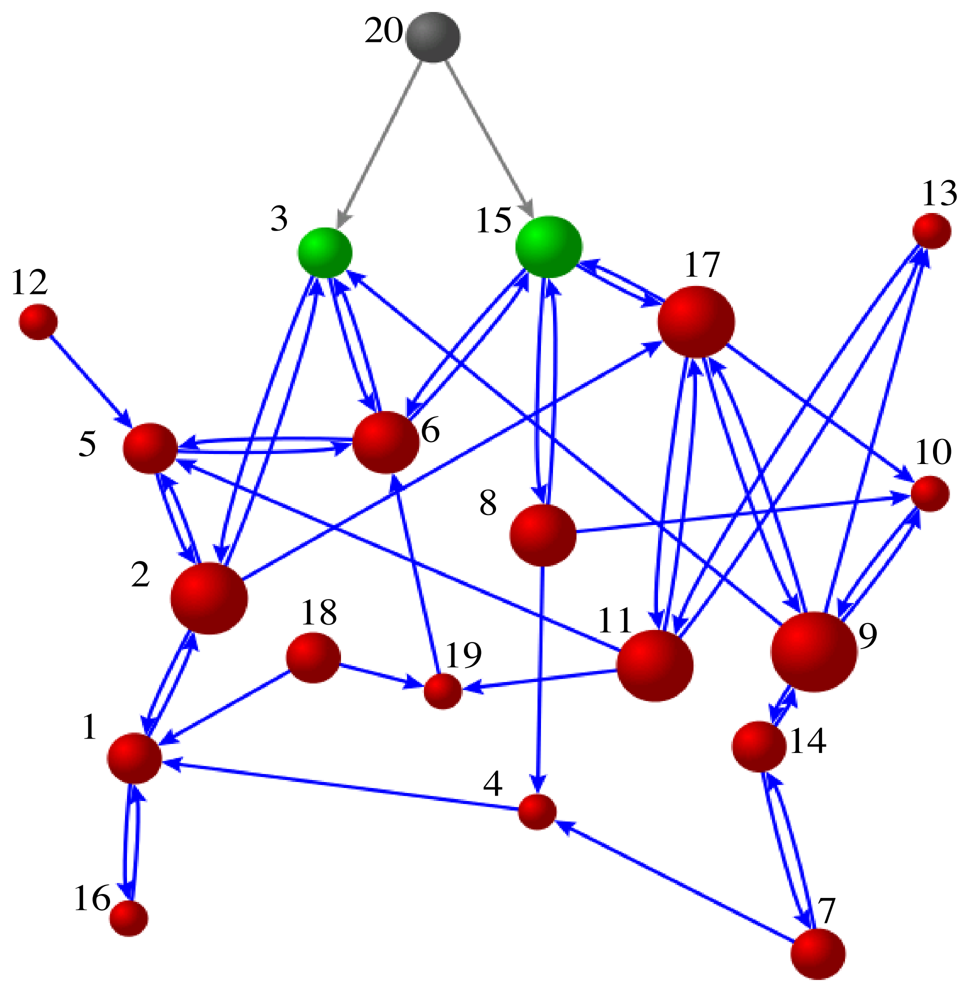
Problem 2: Reverse-engineering of complex networks



Assumption: all nodes are externally accessible

Full network topology?

Problem 3: Detecting hidden nodes



No information is available from the black node. How can we ascertain its existence and its location in the network?

Basic idea (1)

Dynamical system: $d\mathbf{x}/dt = \mathbf{F}(\mathbf{x})$, $\mathbf{x} \in R^m$

Goal: to determine $\mathbf{F}(\mathbf{x})$ from measured time series $\mathbf{x}(t)$!

Power-series expansion of jth component of vector field $\mathbf{F}(\mathbf{x})$

$$[\mathbf{F}(\mathbf{x})]_j = \sum_{l_1=0}^{n} \sum_{l_2=0}^{n} \cdots \sum_{l_m=0}^{n} (a_j)_{l_1 l_2 \ldots l_m} x_1^{l_1} x_2^{l_2} \ldots . x_m^{l_m}$$

x_k – kth component of $\mathbf{x}$; Highest-order power: n

$(a_j)_{l_1 l_2 \ldots l_m}$ - coefficients to be estimated from time series

- $(1+n)^m$ coefficients altogether

If $\mathbf{F}(\mathbf{x})$ contains only a few power-series terms, most of the coefficients will be zero.

W.-X. Wang, R. Yang, Y.-C. Lai, V. Kovanis, and C. Grebogi, *Physical Review Letters* **106**, 154101 (2011).

Basic idea (2)

Concrete example: m = 3 (phase-space dimension): (x,y,z)

n = 3 (highest order in power-series expansion)

total $(1+n)^m = (1+3)^3 = 64$ unknown coefficients

$$[\mathbf{F}(\mathbf{x})]_1 = (a_1)_{0,0,0}x^0y^0z^0 + (a_1)_{1,0,0}x^1y^0z^0 + \ldots + (a_1)_{3,3,3}x^3y^3z^3$$

Coefficient vector $\mathbf{a}_1 = \begin{pmatrix} (a_1)_{0,0,0} \\ (a_1)_{1,0,0} \\ \ldots \\ (a_1)_{3,3,3} \end{pmatrix}$ - 64×1

Measurement vector $\mathbf{g}(t) = [x(t)^0y(t)^0z(t)^0, x(t)^1y(t)^0z(t)^0, \ldots, x(t)^3y(t)^3z(t)^3]$

1×64

So $[\mathbf{F}(\mathbf{x}(t))]_1 = \mathbf{g}(t) \bullet \mathbf{a}_1$

Basic idea (3)

Suppose $\mathbf{x}(t)$ is available at times $t_{0,}t_1,t_2,\ldots,t_{10}$ (11 vector data points)

$$\frac{dx}{dt}(t_1) = [\mathbf{F}(\mathbf{x}(t_1))]_1 = \mathbf{g}(t_1)\bullet\mathbf{a}_1$$

$$\frac{dx}{dt}(t_2) = [\mathbf{F}(\mathbf{x}(t_2))]_1 = \mathbf{g}(t_2)\bullet\mathbf{a}_1$$

$$\ldots$$

$$\frac{dx}{dt}(t_{10}) = [\mathbf{F}(\mathbf{x}(t_{10}))]_1 = \mathbf{g}(t_{10})\bullet\mathbf{a}_1$$

$$\text{Derivative vector } d\mathbf{X} = \begin{pmatrix} (dx/dt)(t_1) \\ (dx/dt)(t_2) \\ \ldots \\ (dx/dt)(t_{10}) \end{pmatrix}_{10\times 1} ; \text{ Measurement matrix } \mathbf{G} = \begin{pmatrix} \mathbf{g}(t_1) \\ \mathbf{g}(t_2) \\ \vdots \\ \mathbf{g}(t_{10}) \end{pmatrix}_{10\times 64}$$

We finally have $d\mathbf{X} = \mathbf{G}\bullet\mathbf{a}_1$ or $d\mathbf{X}_{10\times 1} = \mathbf{G}_{10\times 64}\bullet(\mathbf{a}_1)_{64\times 1}$

Basic idea (4)

$$d\mathbf{X} = \mathbf{G} \bullet \mathbf{a}_1 \qquad \text{or} \qquad d\mathbf{X}_{10\times 1} = \mathbf{G}_{10\times 64} \bullet (\mathbf{a}_1)_{64\times 1}$$

Reminder: $\mathbf{a}_1$ is the coefficient vector for the first dynamical variable x.

To obtain $[\mathbf{F}(\mathbf{x})]_2$, we expand

$$[\mathbf{F}(\mathbf{x})]_2 = (a_2)_{0,0,0} x^0 y^0 z^0 + (a_2)_{1,0,0} x^1 y^0 z^0 + \ldots + (a_2)_{3,3,3} x^3 y^3 z^3$$

with $\mathbf{a}_2$, the coefficient vector for the second dynamical variable y. We have

$$d\mathbf{Y} = \mathbf{G} \bullet \mathbf{a}_2 \qquad \text{or} \qquad d\mathbf{Y}_{10\times 1} = \mathbf{G}_{10\times 64} \bullet (\mathbf{a}_2)_{64\times 1}$$

where

$$d\mathbf{Y} = \begin{pmatrix} (dy/dt)(t_1) \\ (dy/dt)(t_2) \\ \ldots \\ (dy/dt)(t_{10}) \end{pmatrix}_{10\times 1}.$$

Note: the measurement matrix **G** is the same.

Similar expressions can be obtained for all components of the velocity field.

Compressive sensing (1)

Look at

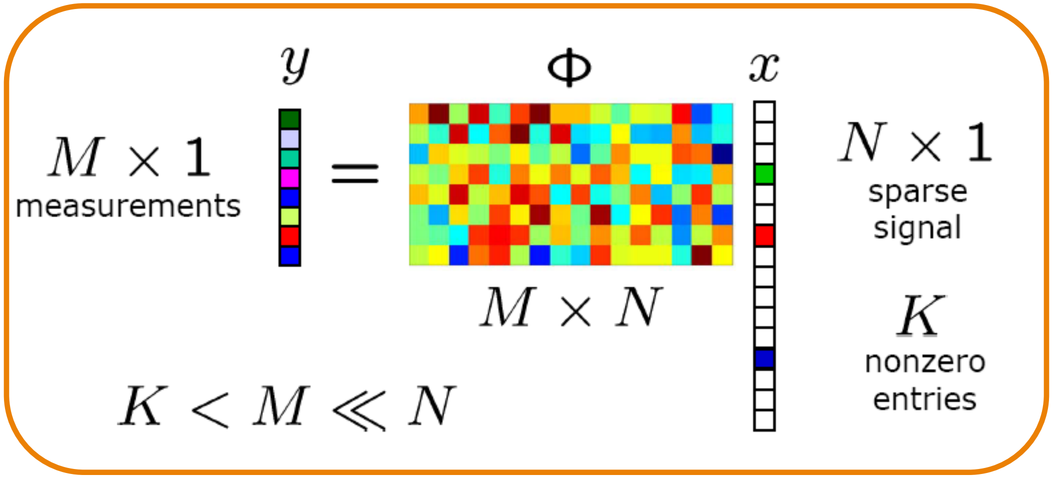
$\mathrm{d}\mathbf{X} = \mathbf{G} \bullet \mathbf{a}_1$ or $\mathrm{d}\mathbf{X}_{10\times 1} = \mathbf{G}_{10\times 64} \bullet (\mathbf{a}_1)_{64\times 1}$

Note that $\mathbf{a}_1$ is sparse - Compressive sensing!

Data/Image compression:

Φ: Random projection (not full rank)

x - sparse vector to be recovered

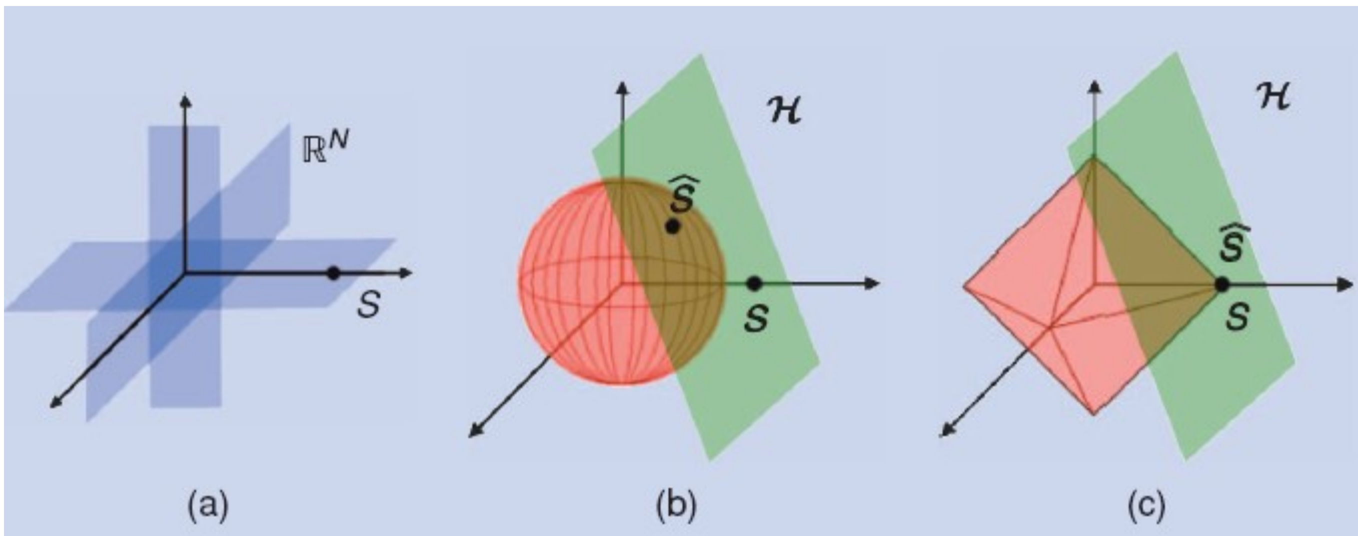


Goal of compressive sensing: Find a vector x with minimum number of entries subject to the constraint $y = \Phi \bullet x$

Compressive Sensing (2)

Find a vector x with minimum number of entries subject to the constraint $y = \Phi \bullet x$: $l_1 -$ norm

Why $l_1 -$ norm? - Simple example in three dimensions



E. Candes, J. Romberg, and T. Tao, *IEEE Trans. Information Theory* **52**, 489 (2006), *Comm. Pure. Appl. Math*. **59**, 1207 (2006);

D. Donoho, *IEEE Trans. Information Theory* **52**, 1289 (2006));

Special review: *IEEE Signal Process. Mag*. **24**, 2008

Predicting catastrophe (1)

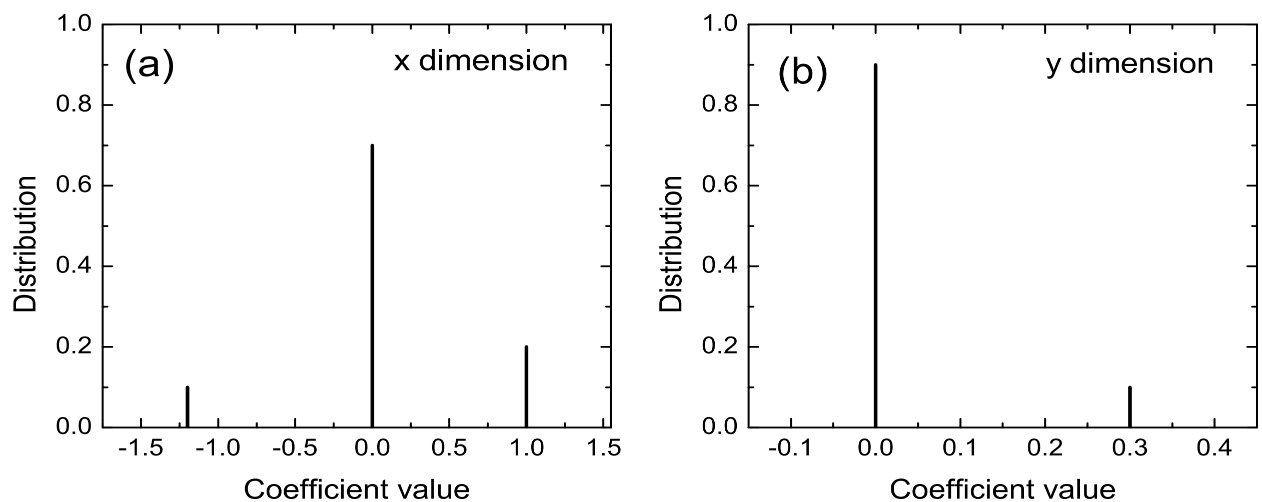
Henon map: $(x_{n+1}, y_{n+1}) = (1 - a x_n^2 + y_n, b x_n)$

Say the system operates at parameter values: $a = 1.2$ and $b = 0.3$.

There is a chaotic attractor.

Can we assess if a catastrophic bifurcation (e.g., crisis) is imminent based on a limited set of measurements?

Step 1: Predicting system equations



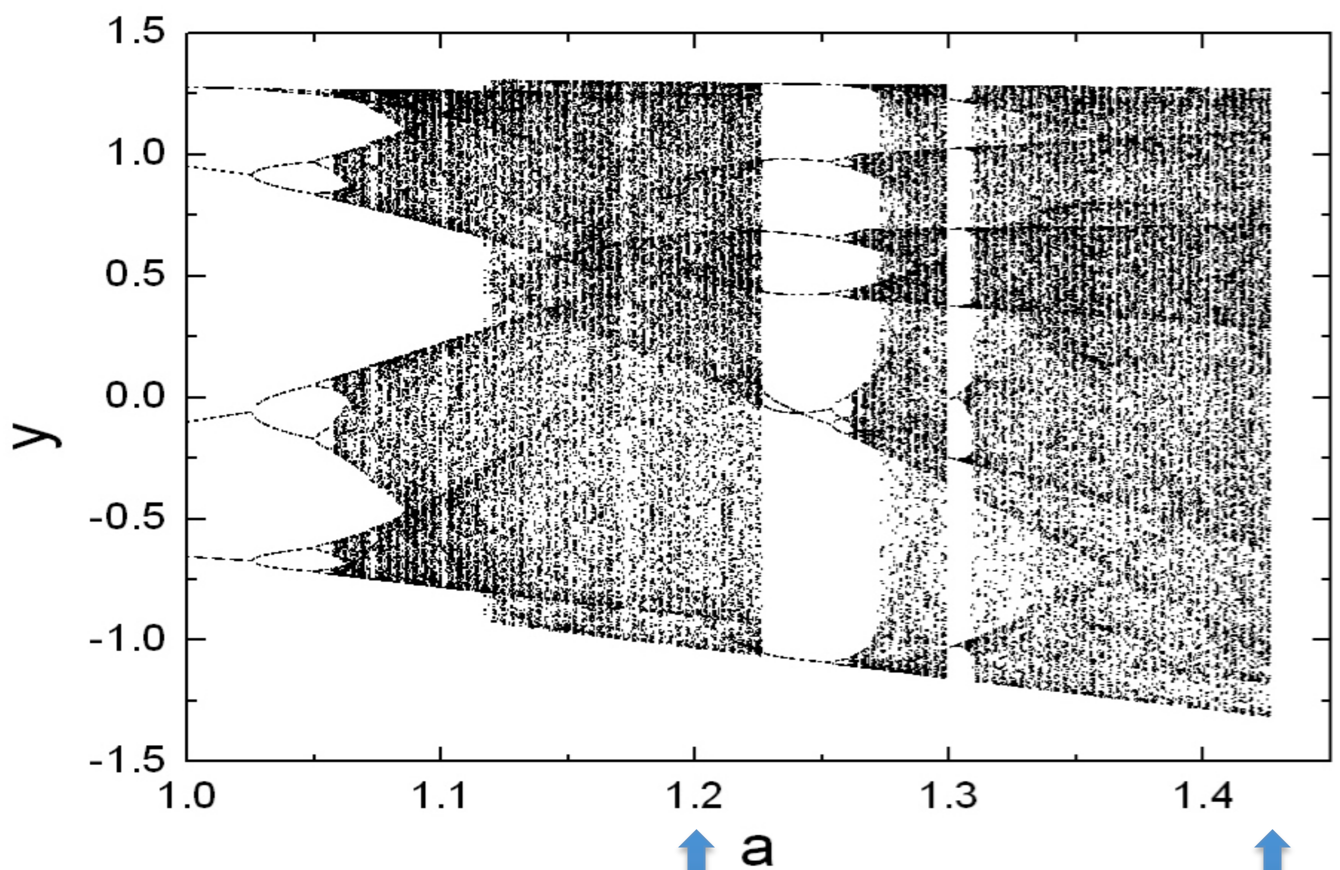
Distribution of predicted values of ten power-series coefficients:

constant, y, y^2, y^3, x, xy, xy^2, x^2, x^2y, x^3

# of data points used: 8

Predicting catastrophe (2)

Step 2: Performing numerical bifurcation analysis

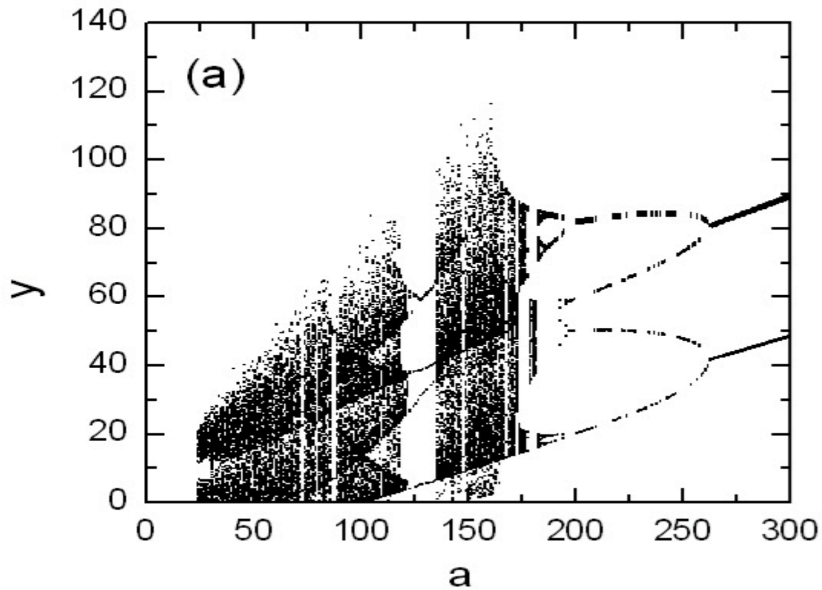


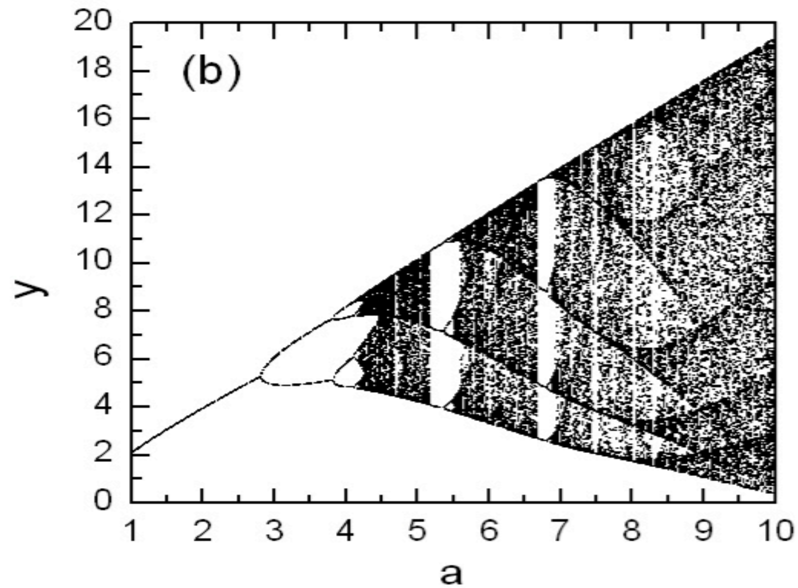
Current operation point

Boundary Crisis

Predicting catastrophe (3)

Examples of predicting continuous-time dynamical systems





Classical Lorenz system

$dx/dt = 10y - 10x$

$dy/dt = x(a - z) - y$

$dz/dt = xy - (2/3)z$

Classical Rossler system

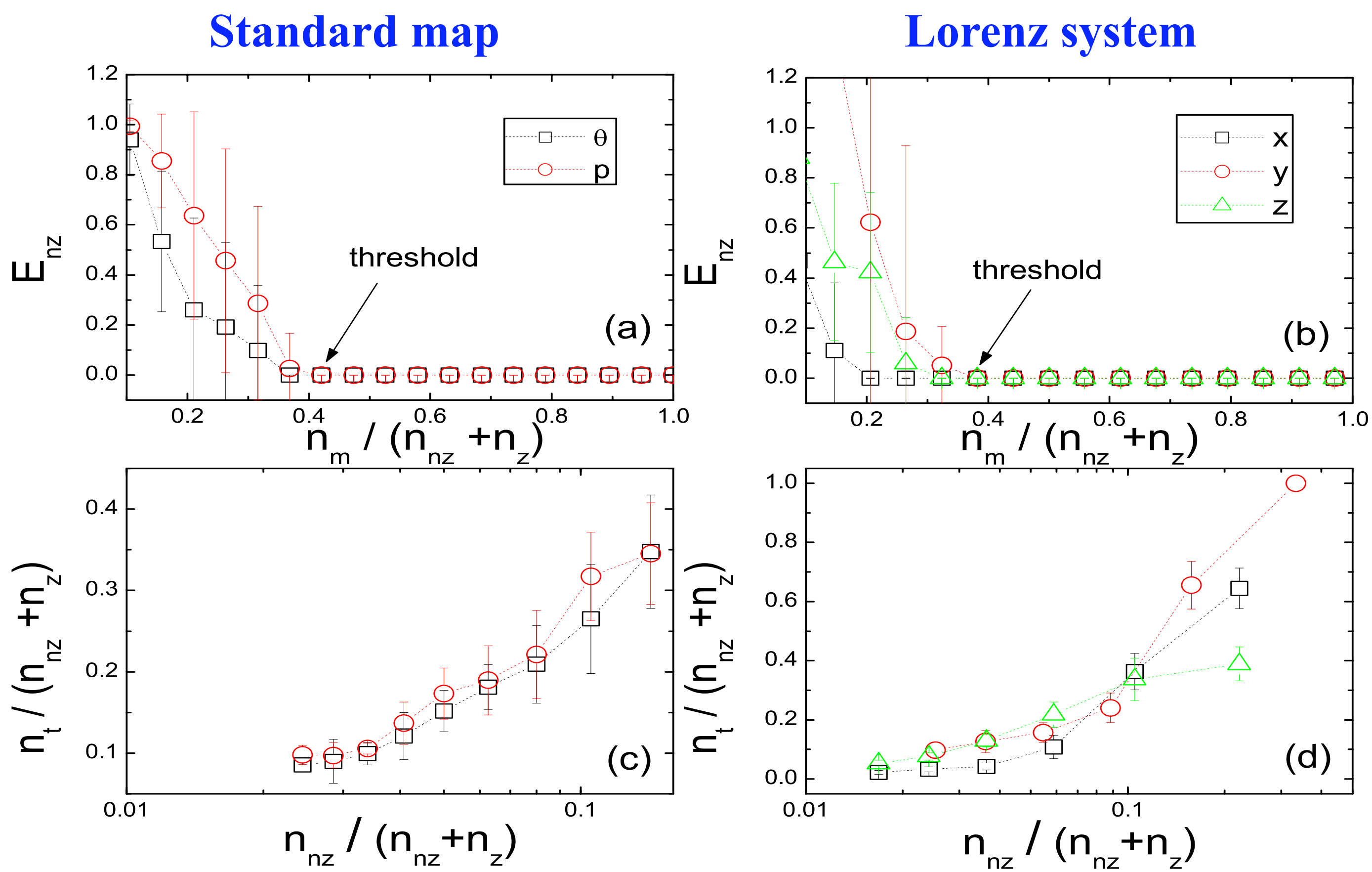
$dx/dt = -y - z$

$dy/dt = x + 0.2y$

$dz/dt = 0.2 + z(x - a)$

# of data points used: 18

Performance analysis



n_m – # of measurements

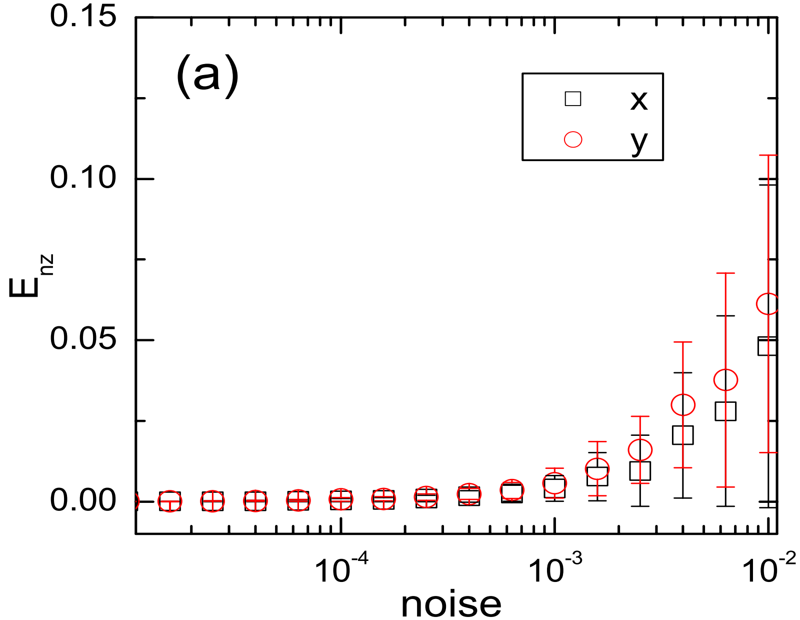
n_{nz} – # of non-zero coefficients; n_z – # of zero coefficients

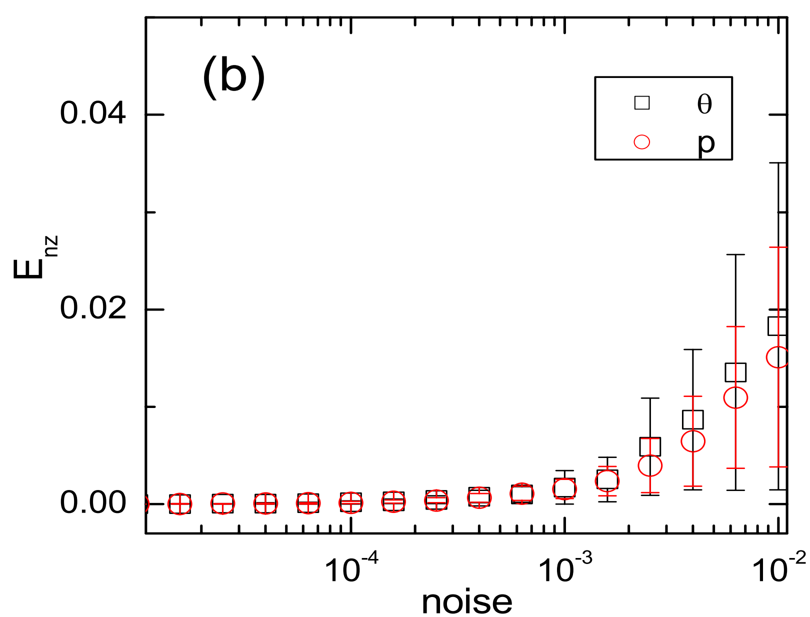
$(n_{nz}+n_z)$ – total # of coefficients to be determined

n_t – minimum # of measurements required for accurate prediction

Effect of noise

Henon map | **Standard map**





$n_m = 8$

$(n_{nz} + n_z) = 16$

$n_m = 10$

$(n_{nz} + n_z) = 20$

W.-X. Wang, R. Yang, Y.-C. Lai, V. Kovanis, and C. Grebogi, *Physical Review Letters* **106**, 154101 (2011).

Predicting future attractors of time-varying dynamical systems (1)

Dynamical system: $d\mathbf{x}/dt = \mathbf{F}[\mathbf{x}, \mathbf{p}(t)]$, $\mathbf{x} \in R^m$

$\mathbf{p}(t)$ - parameters varying slowly with time

T_M – measurement time period;

$\mathbf{x}(t)$ - available in time interval: $t_M - T_M \le t \le t_M$

Goal: to determine both $\mathbf{F}[\mathbf{x}, \mathbf{p}(t)]$ and $\mathbf{p}(t)$ from available time series $\mathbf{x}(t)$ so that the nature of the attractor for $t > t_M$ can be assessed.

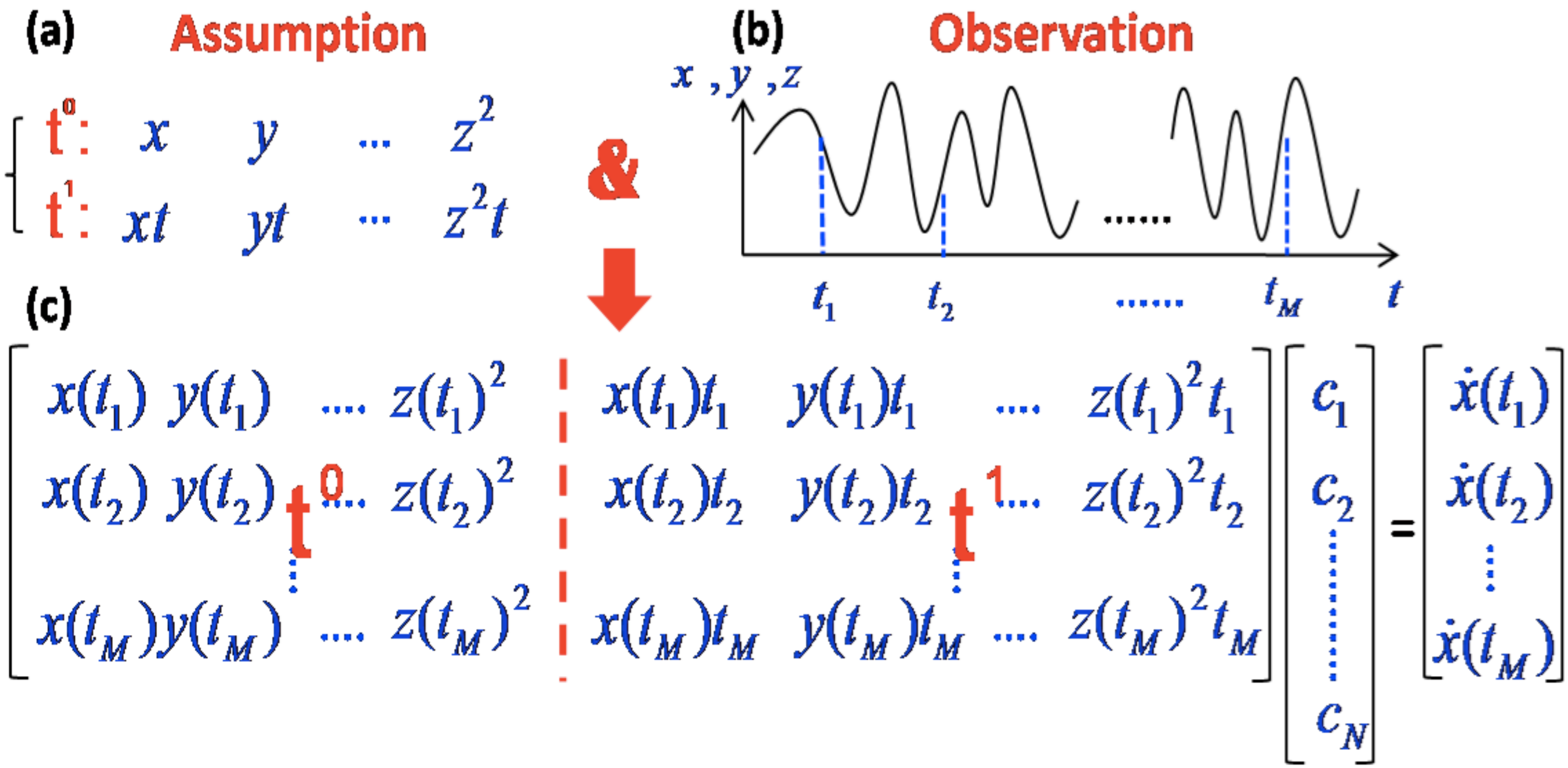
Power-series expansion

$$[\mathbf{F}(\mathbf{x})]_j = \sum_{l_1=0}^{n}\sum_{l_2=0}^{n}\cdots\sum_{l_m=0}^{n}(\alpha_j)_{l_1 l_2 \ldots l_m} x_1^{l_1} x_2^{l_2} \ldots x_m^{l_m} \sum_{w=0}^{v}(\beta_j)_w t^w$$

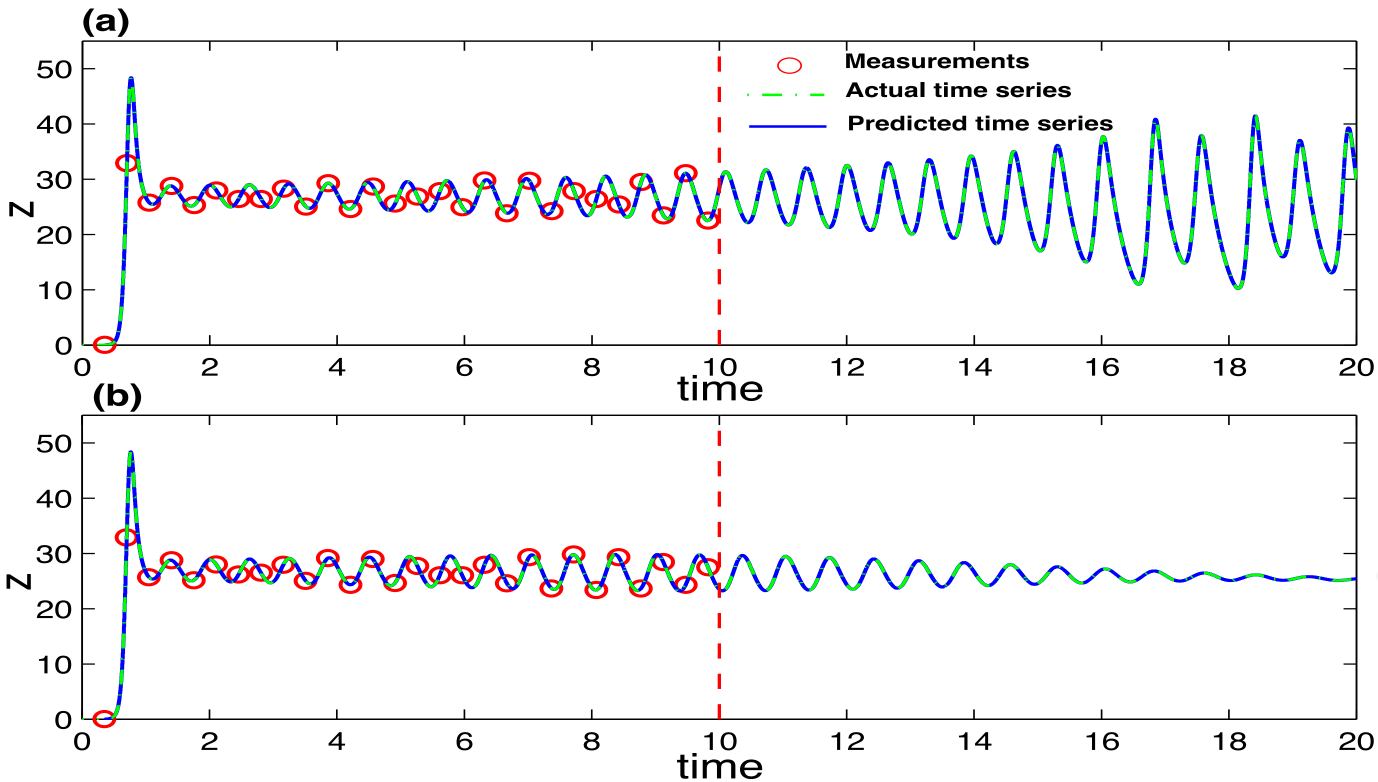
$$= \sum_{l_1,\ldots,l_m=0}^{n}\sum_{w=0}^{v}(c_j)_{l_1,\ldots,l_m;w} x_1^{l_1} x_2^{l_2} \ldots x_m^{l_m} \cdot t^w \leftrightarrow \text{CS framework}$$

Predicting future attractors of time-varying dynamical systems (2)

Formulated as a CS problem:



Predicting future attractors of time-varying dynamical systems (3)



$k_1(t) = 0$

$k_2(t) = 0$

$k_3(t) = 0$

$k_4(t) = 0$

$k_1(t) = -t^2$

$k_2(t) = 0.5t$

$k_3(t) = t$

$k_4(t) = -0.5t^2$

Time-varying Lorenz system

$$dx/dt = -10(x - y) + k_1(t) \cdot y$$

$$dy/dt = 28x - y - xz + k_2(t) \cdot z$$

$$dz/dt = xy - (8/3)z + [k_3(t) + k_4(t)] \cdot y$$

R. Yang, Y.-C. Lai, and C. Grebogi, "Forecasting the future: is it possible for time-varying nonlinear dynamical systems," Chaos 22, 033119 (2012).

Uncovering full topology of oscillator networks (1)

A class of commonly studied oscillator -network models:

$$\frac{d\mathbf{x}_i}{dt} = \mathbf{F}_i(\mathbf{x}_i) + \sum_{j=1, j\neq i}^{N} \mathbf{C}_{ij} \bullet (\mathbf{x}_j - \mathbf{x}_i) \qquad (i = 1, \ldots, N)$$

- dynamical equation of node i

N - size of network, $\mathbf{x}_i \in R^m$, $\mathbf{C}_{ij}$ is the *local* coupling matrix

$$\mathbf{C}_{ij} = \begin{pmatrix} C_{ij}^{1,1} & C_{ij}^{1,2} & \cdots & C_{ij}^{1,m} \\ C_{ij}^{2,1} & C_{ij}^{2,2} & \cdots & C_{ij}^{2,m} \\ \vdots & \vdots & \vdots & \vdots \\ C_{ij}^{m,1} & C_{ij}^{m,2} & \cdots & C_{ij}^{m,m} \end{pmatrix}$$

- determines full topology

If there is at least one nonzero element in $\mathbf{C}_{ij}$, nodes i and j are coupled.

Goal: to determine all $\mathbf{F}_i(\mathbf{x}_i)$ and $\mathbf{C}_{ij}$ from time series.

Uncovering full topology of oscillator networks (2)

$$\mathbf{X} = \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \cdots \\ \mathbf{x}_N \end{pmatrix}_{Nm\times 1}$$

– Network equation is $\frac{d\mathbf{X}}{dt} = \mathbf{G}(\mathbf{X})$, where

$$[\mathbf{G}(\mathbf{X})]_i = \mathbf{F}_i(\mathbf{x}_i) + \sum_{j=1, j\neq i}^{N} \mathbf{C}_{ij} \bullet (\mathbf{x}_j - \mathbf{x}_i)$$

- A very high-dimensional (*Nm*-dimensional) dynamical system;
- For complex networks (e.g, random, small-world, scale-free), node-to-node connections are typically sparse;
- In power-series expansion of $[\mathbf{G}(\mathbf{X})]_i$, most coefficients will be zero - guaranteeing sparsity condition for compressive sensing.

W.-X. Wang, R. Yang, Y.-C. Lai, V. Kovanis, M. A. F. Harrison, "Time-series based prediction of complex oscillator networks via compressive sensing", *Europhysics Letters* **94**, 48006 (2011).

Evolutionary-game dynamics

Example:
Prisoner's dilemma game

	Cooperate	Defect
Cooperate	win-win	lose much-win much
Defect	win much-lose much	lose-lose

Strategies: cooperation $\mathbf{S}(C) = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$; defection $\mathbf{S}(D) = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$

Payoff matrix: $\mathbf{P}(PD) = \begin{pmatrix} 1 & 0 \\ \mathrm{b} & 0 \end{pmatrix}$ b - parameter

Payoff of agent x from playing PDG with agent y: $\boxed{\mathrm{M}_{x \leftarrow y} = \mathbf{S}_x^T \mathbf{P} \mathbf{S}_y}$

For example, $\mathrm{M}_{\mathrm{C} \leftarrow \mathrm{C}} = 1$

$\mathrm{M}_{\mathrm{D} \leftarrow \mathrm{D}} = 0$

$\mathrm{M}_{\mathrm{C} \leftarrow \mathrm{D}} = 0$

$\mathrm{M}_{\mathrm{D} \leftarrow \mathrm{C}} = \mathrm{b}$

Evolutionary game on network (social and economical networks)

A network of agents playing games with one another:

$$\text{Adjacency matrix} = \begin{pmatrix} \cdots & \cdots & \cdots \\ \cdots & a_{xy} & \cdots \\ \cdots & \cdots & \cdots \end{pmatrix} : \begin{cases} a_{xy} = 1 & \text{if } x \text{ connects with } y \\ a_{xy} = 0 & \text{if no connection} \end{cases}$$

Payoff of agent x from agent y: $\mathrm{M}_{x \leftarrow y} = a_{xy} \mathbf{S}_x^T \mathbf{PS}_y$

Time series of agents (Detectable) $\begin{cases} \text{(1) payoffs} \\ \text{(2) strategies} \end{cases}$ $\xrightarrow{\text{Compressive sensing}}$ Full social network structure

Prediction as a CS Problem

Payoff of x at time t: $M_x(t) = a_{x1}\mathbf{S}_x^T(t)\mathbf{P}\mathbf{S}_1(t) + a_{x2}\mathbf{S}_x^T(t)\mathbf{P}\mathbf{S}_2(t) + \cdots + a_{xN}\mathbf{S}_x^T(t)\mathbf{P}\mathbf{S}_N(t)$

$$\mathbf{Y} = \begin{pmatrix} M_x(t_1) \\ M_x(t_2) \\ \vdots \\ M_x(t_m) \end{pmatrix} \qquad \mathbf{X} = \begin{pmatrix} a_{x1} \\ a_{x2} \\ \vdots \\ a_{xN} \end{pmatrix}$$

$\mathbf{X}$: connection vector of agent x (to be predicted)

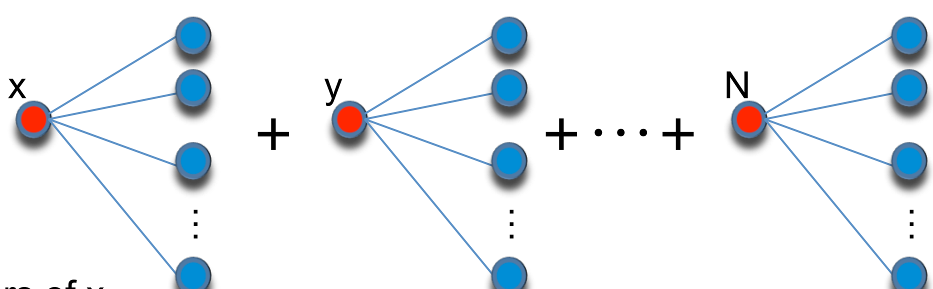
$$\Phi = \begin{pmatrix} \mathbf{S}_x^T(t_1)\mathbf{P}\mathbf{S}_1(t_1) & \mathbf{S}_x^T(t_1)\mathbf{P}\mathbf{S}_2(t_1) & \cdots & \mathbf{S}_x^T(t_1)\mathbf{P}\mathbf{S}_N(t_1) \\ \mathbf{S}_x^T(t_2)\mathbf{P}\mathbf{S}_1(t_2) & \mathbf{S}_x^T(t_2)\mathbf{P}\mathbf{S}_2(t_2) & \cdots & \mathbf{S}_x^T(t_2)\mathbf{P}\mathbf{S}_N(t_2) \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{S}_x^T(t_m)\mathbf{P}\mathbf{S}_1(t_m) & \mathbf{S}_x^T(t_m)\mathbf{P}\mathbf{S}_2(t_m) & \cdots & \mathbf{S}_x^T(t_m)\mathbf{P}\mathbf{S}_N(t_m) \end{pmatrix}$$

W.-X. Wang, Y.-C. Lai, C. Grebogi, and J.-P. Ye, "Network reconstruction based on evolutionary-game data," *Physical Review X* 1, 021021 (2011).

$\boxed{\mathbf{Y} = \Phi \mathbf{X} \qquad \mathbf{Y}, \Phi\text{: obtainable from time series}}$

Compressive sensing

$$\mathbf{X} = \begin{pmatrix} a_{x1} \\ a_{x2} \\ \vdots \\ a_{xN} \end{pmatrix} \Leftrightarrow$$

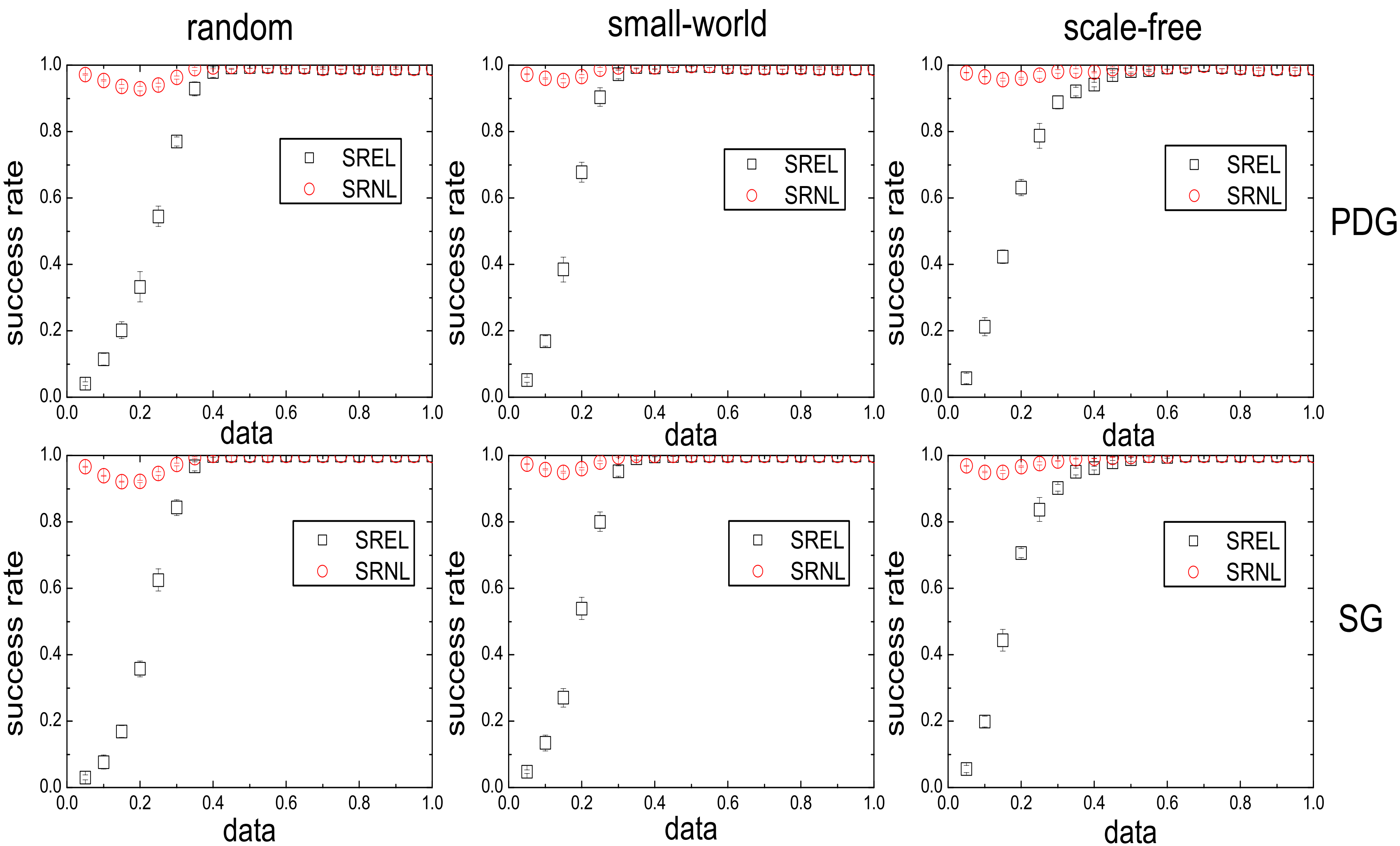


Neighbors of x

matching

Full network structure

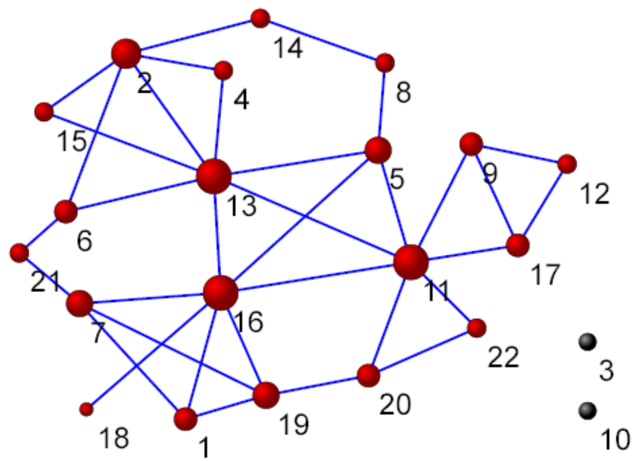
Success rate for model networks



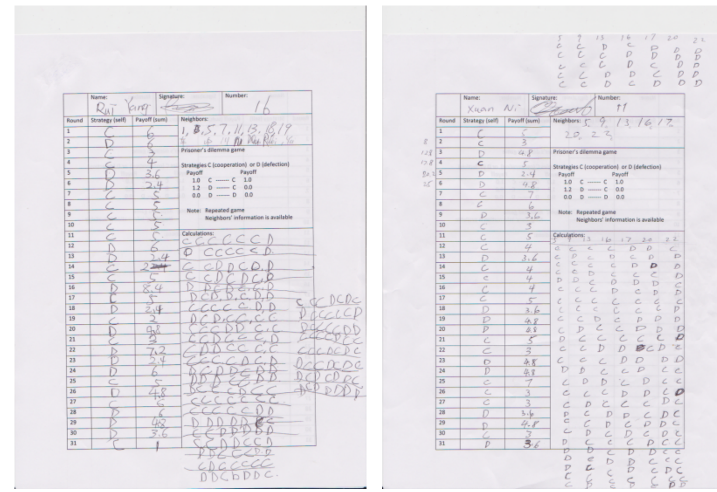
Reverse engineering of a real social network

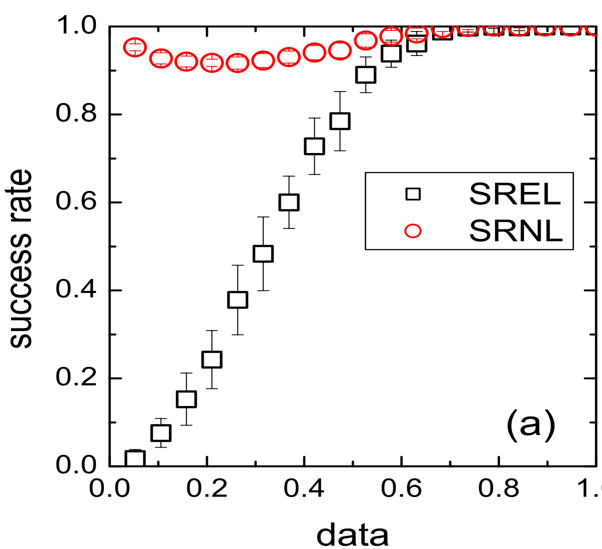
22 students play PDG together and
write down their payoffs and strategies

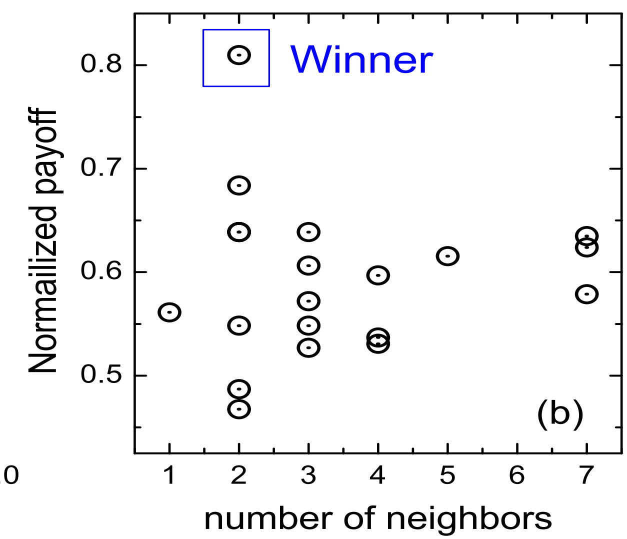
Friendship network



Experimental record of two players

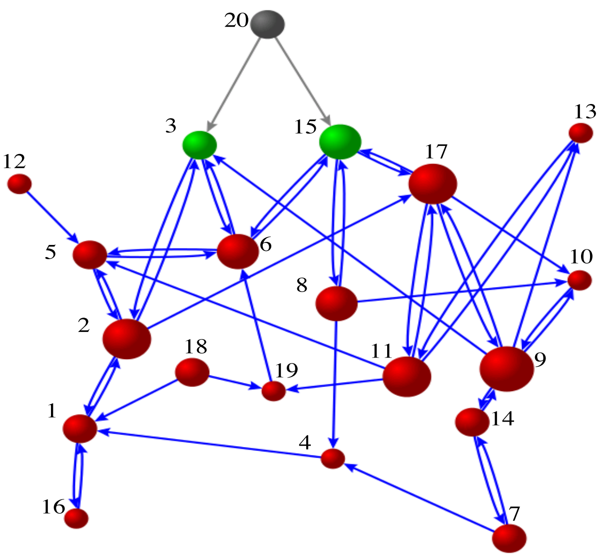


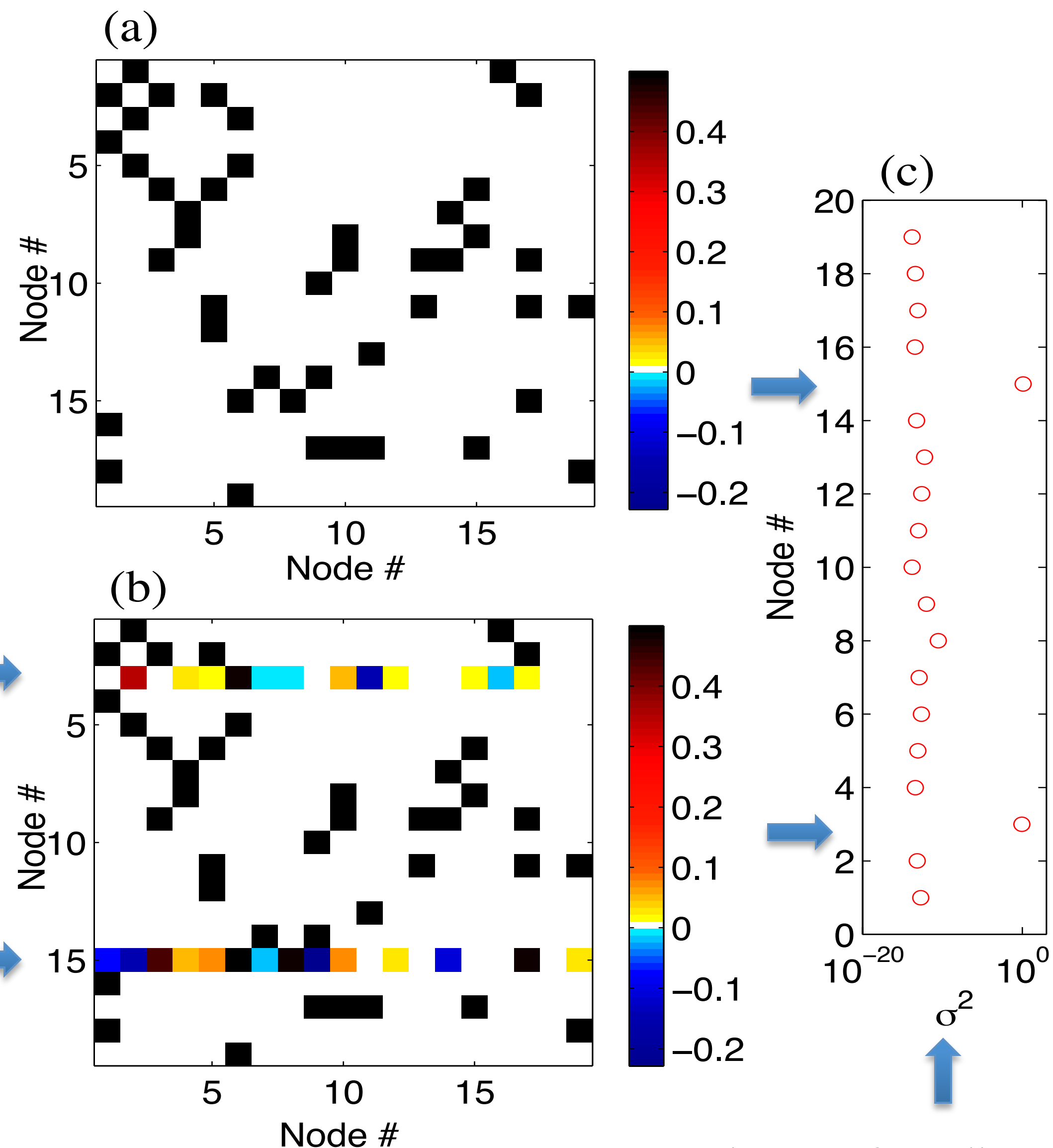




Observation:
Large-degree nodes are not necessarily winners

Detecting hidden nodes





Idea

- Two green nodes: immediate neighbors of hidden node
- Information from green nodes is not complete
- Anomalies in the prediction of connections of green nodes

Directed/weighted network: adjacency matrix not symmetric

Variance of predicted coefficients

Discussion (1)

1. Key requirement of compressive sensing – the vector to be determined must be sparse.

   Dynamical systems - three cases:

- Vector field/map contains a few Fourier-series terms - Yes
- Vector field/map contains a few power-series terms - Yes
- Vector field /map contains many terms – not known

Ikeda Map: $F(x,y) = [A + B(x\cos\phi - y\sin\phi), B(x\sin\phi + y\cos\phi)]$

where $\phi \equiv p - \dfrac{k}{1 + x^2 + y^2}$ - describes dynamics in an optical cavity

Mathematical question: given an arbitrary function, can one find a suitable base of expansion so that the function can be represented by a limited number of terms?

Discussion (2)

2. Networked systems described by evolutionary games – Yes
3. Measurements of ALL dynamical variables are needed.

   Outstanding issue

   If this is not the case, say, if only one dynamical variable can be measured, the CS-based method would not work.

   Delay-coordinate embedding method?

   - gives only a topological equivalent of the underlying dynamical system (e.g., Takens' embedding theorem guarantees only a one-to-one correspondence between the true system and the reconstructed system).
4. In **Conclusion**, much work is needed!