How Entropic Regression Beats the Outliers Problem in Nonlinear System Identification: Supplementary Information

Abd AlRahman R. AlMomani[1,5], Jie Sun[1,2,3,4], and Erik Bollt[1,2,4,5]

[1]Clarkson Center for Complex Systems Science (C^3S^2), Potsdam, NY, USA
[2]Department of Mathematics, Clarkson University, Potsdam, NY, USA
[3]Department of Computer Science, Clarkson University, Potsdam, NY, USA
[4]Department of Physics, Clarkson University, Potsdam, NY, USA
[5]Department of Electrical and Computer Engineering, Clarkson University, Potsdam, NY, USA

Contents

1 Governing Dynamics, Over-sparsity, and Sensitivity for Expansion Order

In the main text we discussed the effect of polynomial expansion order choses to construct the basis matrix Φ, and in this section we provide numerical excrements to show the effect of polynomial expansion order.

Recall from our main text Eq.(26), in Compressive Sensing (CS) framework we solves the constrained optimization problem:

$$\begin{cases} \arg\min_{\boldsymbol{a}} \|\boldsymbol{a}\|_1, \\ \text{subject to } \|\Phi\boldsymbol{a} - \boldsymbol{f}\| \leq \epsilon, \end{cases} \tag{1}$$

where the parameter $\epsilon \geq 0$ is used to relax the otherwise strict constraint $\Phi\boldsymbol{a} = \boldsymbol{f}$, to allow for the presence of noise in data.

Fig. 1 shows the reconstructed model by CS for the first equation of Lorenz system regarding $\dot{x}$. We observe sensitivity on expansion order and how CS over-sparse the solution at with the 7^{th} order expansion. This results was with using 300 measurements, and to extend the investigation we repeat the experiment for the 7th expansion order with doubled number of measurements, and Fig. 2 shows that CS still over-sparse the solution. This shows the relative sensitivity of CS with respect to the expansion order of basis functions.

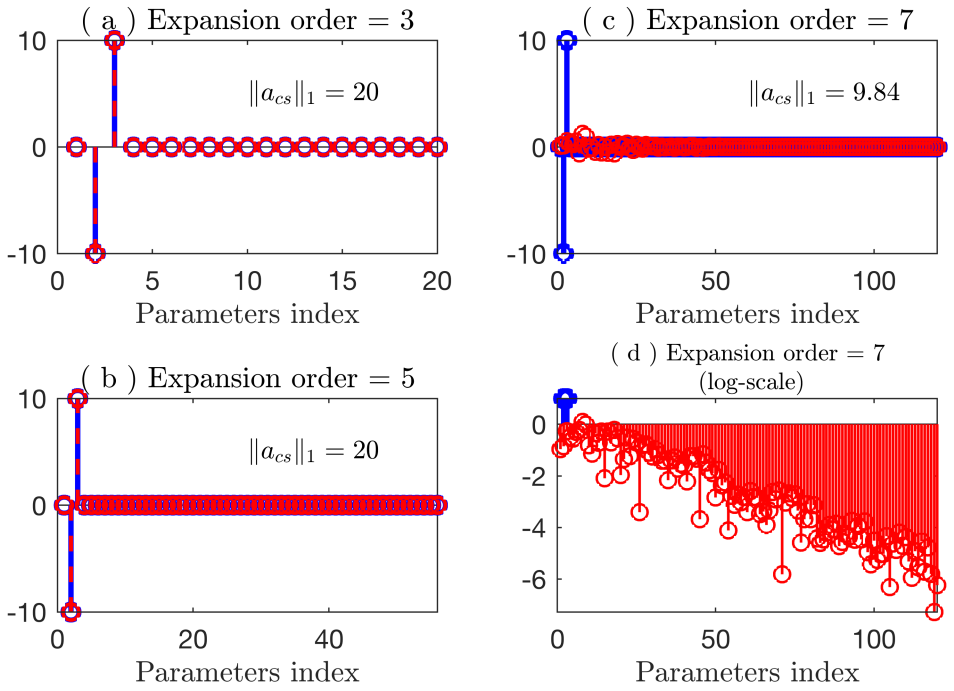


Figure 1: CS reconstructed model for the first equation of Lorenz system regarding $\dot{x}$. The Solution shown in $\log_{10}$-scale in the y-axis for the parameters magnitude. From left to right, we see the recovered solution using the 3^{rd}, 5^{th} and 7^{th} expansion order respectively. We used 300 noise free measurement, ($\epsilon_1 = \epsilon_2 = 0$). We see that with the 3^{rd} order polynomial expansion, CS recovered the solution with high accuracy, and it the same case with 5^{th} order polynomial expansion although the accuracy is slightly reduced, but we can still see the accurate sparse structure clearly. With the 7^{th} order polynomial expansion which produce 120 basis, we see a complete failure of CS where it over sparse the solution to have $\|\boldsymbol{a}_{cs}\|_1 = 0.005$.

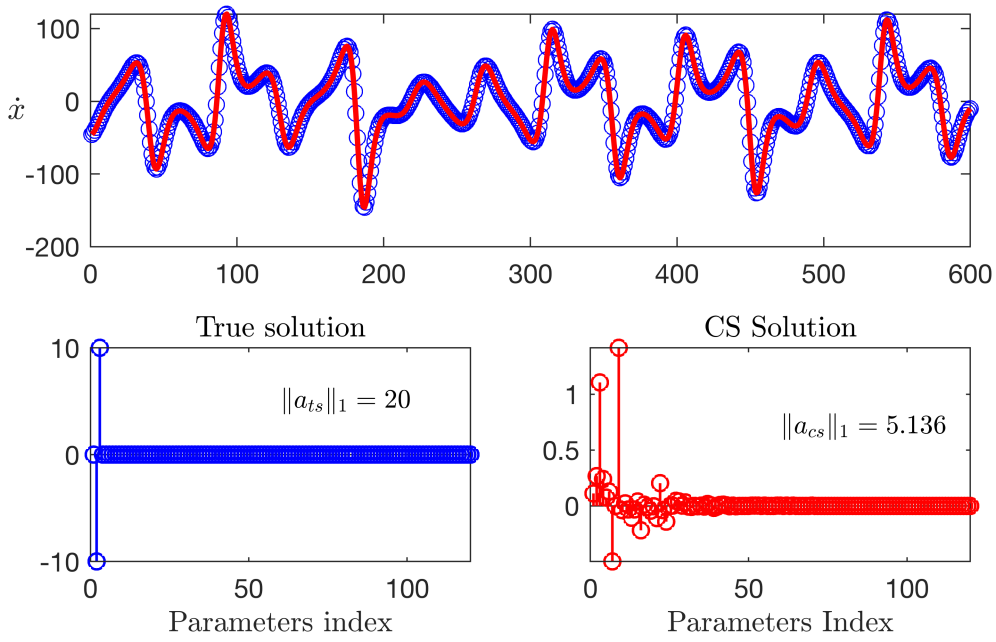


Figure 2: CS Recovered solution for the first term $\dot{x}$ of Lorenz system using 600 noise free measurement, ($\epsilon_1 = \epsilon_2 = 0$). We see that even when we doubled the measurements, the CS is still over-sparse, although we have a good fitting curve, but the recovered system is wrong. In the other hand, the CS performs poor in recovering such dynamic with the presence of noise even with considering a low expansion order. Click here for a simulation of CS results of the same current example with considering the 3^{rd} order polynomial expansion and the presence of noise.

In order to construct a second example that clearly shows the oversparse mechanism in CS, consider the three-dimensional linear system:

$$\begin{pmatrix} 6 & 3 & 2 \\ 2 & 1 & 1 \\ 1 & 2 & 1 \end{pmatrix} \boldsymbol{a} = \begin{pmatrix} 6 \\ 2 \\ 4 \end{pmatrix}. \tag{2}$$

It is easy to find that the solution for the above system is $\boldsymbol{a} = \begin{pmatrix} 0 & 2 & 0 \end{pmatrix}^T$. Now, suppose that the third "measurement" is missing, and we have the under-determined system

$$\begin{pmatrix} 6 & 3 & 2 \\ 2 & 1 & 1 \end{pmatrix} \boldsymbol{a} = \begin{pmatrix} 6 \\ 2 \end{pmatrix} \tag{3}$$

then we have infinitely many solutions lies on the line of intersection of the two planes:

$$\begin{cases} 6x + 3y + 2z &= 6 \\ 2x + y + z &= 2 \end{cases}$$

Figure 3 shows this simple example, where the solution for $\boldsymbol{a}$ lies on the intersection of the two planes shown, and we see the true solution, the LS solution and CS solution on the solution line. We see how the least square solution is far from the true solution with a high margin of error, but we also see that it only invest in x and y direction where the line of intersections of the two plans lies, then, LS ignore the z direction and

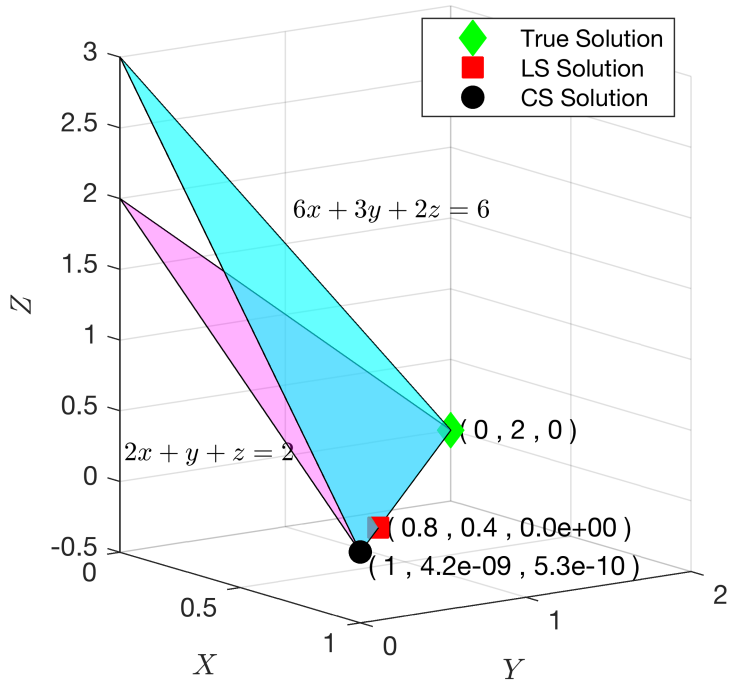


Figure 3: Oversparsity: The line of intersection of the two planes (triangles) shows the solution plane. We see that compressed sensing solution is oversparsed.

try to invest in all feasible directions to reach the best residual.

CS have different mechanism, since within all feasible solutions, it tends to select the one with minimum $\|\boldsymbol{a}\|_1$, even if there is another solution with the same number of sparse that have a residual $\|A\boldsymbol{a} - b\|_2 = 0$, and it is the case in our example where $\|A \begin{pmatrix} 0 & 2 & 0 \end{pmatrix}^T - b\|_2 = 0$, while the CS solution has the residual $\|A\boldsymbol{a}_{CS} - b\|_2 = 2.5 \times 10^{-5}$. In other words, for the system $A\boldsymbol{a} = b$, if there exist two solutions such that $\|\boldsymbol{a_1}\|_1 < \|\boldsymbol{a_2}\|_1$ and $\|A\boldsymbol{a_2} - b\|_2 < \|A\boldsymbol{a_1} - b\|_2 \leq \epsilon$, where ϵ is the tolerance for CS optimization, then CS will select a_1 as a solution, even it has higher residual, and regardless of the structure of the sparse or the information flow between the basis functions and the observations. Numerically, assume the system in Eq. 3 to be:

$$\begin{pmatrix} (6 + 1e^{-10}) & 3 & 2 \\ 2 & (1 + 1e^{-16}) & 1 \end{pmatrix} \boldsymbol{a} = \begin{pmatrix} 6 \\ 2 \end{pmatrix} \tag{4}$$

and consider a reasonable tolerance for CS solver to be $\epsilon = 1e^{-9}$, then CS will always pick [1 0 0] as a solution even it have higher residual. In our ER package, we provide Matlab script (*csdemo.m*) to demonstrate the above example.

For many applications..., it is accepted to have such solution since it lies on the solution line and such residual difference will have negligible effect on the final result, But in discovering the governing equations of dynamical systems, such solution can often lead to a completely wrong structure of the system.

2 Information Theory

The basic idea of information theory can simplified by considering the everyday learning process in our minds. The more information we have about specific topic, the less "new" information we may find in the following days, and less probability to find information resources that can *influence* you with the *new* information that updates yours. In other words and in terms of events occurrence, if the event A has high probability to happens in our daily life, then there will be no (or less) surprise to see the event A occurs. In the other hand, seeing the event B happens which is rare event with low occurrence probability will be "surprise" for us.

We can see the "surprise" term as an indicator of the uncertainty, back to our learning process example, the less the one be surprised about informations he receive the more "certain" he is about the topic he is learning. More surprise indicate higher uncertainty. That leads us to the first subsection about the very basic measure in the information theory which is the Entropy.

2.1 Entropy

Entropy is firstly known as an extensive property of a thermodynamic system. The entropy of a thermodynamic system is a function of the number of possible microscopic states consistent with the macroscopic quantities that characterize the system. Assuming equal probability of the microscopic states, the entropy is given by:

$$S = k_B \ln(W) \tag{5}$$

where W is the number of microscopic states and k_B is Boltzmann constant named after Ludwig Eduard Boltzmann where the Eq. 5 curved on his gravestone. Boltzmann saw entropy as a measure of statistical disorder in the system.

An analog to thermodynamic entropy is information entropy introduced by Claude Shannon in 1948 as "measures of information, choice, and uncertainty". To describe Shannon's entropy, consider a discrete random variable X whose probability mass function is denoted by $p(x) = Prob(X = x)$. One can calculate its entropy as [3, 18],

$$H(X) = -K \sum_x p(x) \log p(x), \tag{6}$$

where K is positive constant, and $H(X)$ is a measure of the uncertainty or unpredictability of X. Note that if we assume uniform probability distribution for the states of X, then we have $p(x) = \frac{1}{N}$, where N is the number of states, and then Eq. 6 can be written as $H(X) = K \log(N)$ similar to Boltzmann's entropy under the same assumption of equal probability of the states. The constant K, as Shannon sates, is merely amounts to a choice of a unit of measurement, and we consider $K = 1$ for the rest of this document for simplicity. Fig. 4 shows the entropy function for a random event with different probability.

Shannon's work provided extended and generalized view and understanding for the entropy, and one of the extended perspectives of Shannon's entropy is dealing with the continuous random variables, and it takes the form:

$$H(X) = \int_{-\infty}^{\infty} f_X(x) \log(f_X(x)) dx, \tag{7}$$

where $f_X(x)$ is the probability density function. The entropy shown in Eq. (7) is referred to the *differential entropy*.

2.2 Mutual Information

The entropy defined in Eq. (6) naturally extends to the case of multiple random variables. For example, the joint entropy $H(X, Y)$, and conditional entropy $H(X|Y)$ of two random variables X and Y is given, respectively, by [3, 18],

$$H(X, Y) = -\sum_{x,y} p(x, y) \log p(x, y) \tag{8}$$

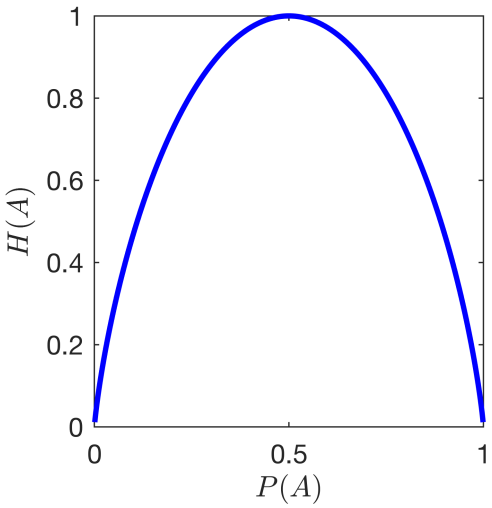


Figure 4: Entropy of the event A. Here we assume the states to be the occurrence and non-occurrence of the event A, and $P(A)$ represent the probability of the occurrence state. This figure show the uncertainty about the event A occurrence. In x-axis we have the probability $P(A) = p$ that the event A occurs, then by Eq. 6 and considering the log to base 2, $H(A) = -p\log(p) - (1-p)\log(1-p)$ is the measure of uncertainty of the event A, where $(1-p)$ is the probability that the event A will not occur. Starting from $P(A) = 1$, meaning that the event A is always occurs or it is the only event we have, then $H(A) = 0$, meaning that there is no uncertainty and we are sure of the event A occurrence. As the probability decrease, the entropy (uncertainty) increase to reach its maximum at $P(A) = 0.5$. Continuing decreasing $P(A)$ will reduce the entropy again, since we become more certain that the event A will not occur, until we become completely certain that A will not occur with $H(A) = 0$ at $P(A) = 0$.

$$\begin{aligned} H(X|Y) &= -\sum_{y} p(y) H(Y|X = x) \\ &= -\sum_{x,y} p(x,y) \log p(x|y), \end{aligned} \tag{9}$$

where $p(x, y)$ is the joint probability distribution, and $H(X|Y)$ (read as entropy of X given Y) is the measure of the uncertainty in X if Y is known. Some of the main properties of the entropy, joint entropy, and conditional entropy can be summarize as follows:

- The entropy of a discrete variable X is positive ($H(X) \geq 0$), while the differential entropy does not satisfy this property.
- For two independent random variables X and Y, $H(X, Y) = H(X) + H(Y)$.
- The chain rule: $H(X, Y) = H(X) + H(Y|X)$.
- One important property is that for a random variable X, the conditional entropy of X given any other variable Y will reduce the entropy of X, meaning that $H(X) \geq H(X|Y)$. The equality holds when X and Y are independent with $H(X, Y) = 0$. This property tells that the information comes from Y reduces the uncertainty about X, and when $Y = X$, that means we have given all the information about X, and then we become completely certain about X, and that gives $H(X|X) = 0$.

The joint and conditional entropies can lead to a measures that detect the statistical dependence or independence between random variables. Such measure is called the mutual information between X and Y,

and it is given by [3, 18],

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X) \\ &= H(X) + H(Y) - H(X,Y), \end{aligned} \tag{10}$$

where the mutual information $I(X;Y)$ (reads as mutual information between X and Y) is a measure of the mutual dependence between the two variables. In terms of joint probability distribution, mutual information can be written as,

$$I(X;Y) = \sum_{y \in Y} \sum_{x \in X} p(x,y) \log \left(\frac{p(x,y)}{p(x)p(y)} \right), \tag{11}$$

and in its continuous form,

$$I(X;Y) = \int_Y \int_X f_{X,Y}(x,y) \log \left(\frac{f_{X,Y}(x,y)}{f_X(x) f_Y(y)} \right), \tag{12}$$

where $f_{X,Y}(x,y)$ is the joint probability density function for the two continuous random variables X and Y.

In case of independence of the two random variables, we have

$$p(x,y) = p(x)p(y), \tag{13}$$

and then we have

$$\log \left(\frac{p(x,y)}{p(x)p(y)} \right) = \log(1) = 0 \implies I(X;Y) = 0. \tag{14}$$

The same principle holds for the continuous variables in Eq. (12), while $I(X;Y)$ satisfy the inequality $I(X;Y) \leqslant min[H(X), H(Y)]$ only in the discrete variables case.

2.3 Transfer Entropy and Causation Entropy

For two stochastic processes X_t and Y_t, the reduction of uncertainty about X_{t+1} due to the information of the past τ_Y states of Y , represented by

$$Y^{(\tau_Y)} = (Y_t, Y_{t-1}, ..., Y_{t-\tau_Y+1}),$$

in addition to the information of the past τ_X states of X , represented by

$$X^{(\tau_X)} = (X_t, X_{t-1}, ..., X_{t-\tau_X+1}),$$

this reduction of uncertainty about X_{t+1} is measured by "Transfer Entropy" which given by [3, 21],

$$T_{Y \to X} = H(X_{t+1}|Xt^{\tau_X}) - H(X_{t+1}|Xt^{\tau_X}, Y_t^{\tau_Y}). \tag{15}$$

The traditional approach of inferring causality between two stochastic processes is to perform the Granger causality test [5]. The main limitation of this test is that it can only provide information about linear dependence between two processes, and therefore fails to capture intrinsic nonlinearities that are common in real-world systems. To overcome this difficulty, Schreiber developed the concept of transfer entropy between two processes [17]. Transfer entropy measures the uncertainty reduction in inferring the future state of a process by learning the (current and past) states of another process.

In our work [21, 20], we showed by several examples that causal relationship inferred by transfer entropy are often misleading when the underlying system contains indirect connections, a dominance of neighboring dynamics, or anticipatory couplings. For example, referring to the main text and the equation $\boldsymbol{f} = \Phi \boldsymbol{a}$, we see that the approaches that consider the transfer entropy in order to find the weak terms in Φ that has no influence on $\dot{X}$ to construct the sparse matrix $\boldsymbol{a}$, these approaches neglect a simple and clear idea that

the terms of Φ has an indirect influence on $\boldsymbol{f}$ through the other terms of Φ. To account for these effects, we developed a measure called *Causation Entropy* (CSE) [21, 20], and show that its appropriate application reveals true coupling structures of the underlying dynamics.

Consider a stochastic network of N processes (nodes) denoted by:

$$X_t = \{X_t^{(1)}, X_t^{(2)}, \ldots, X_t^{(N)}\} \tag{16}$$

where $X_t^{(i)} \in \mathbb{R}^d$ is a random variable representing the state of process (or node) i at time t, and $i \in \mathcal{V} = \{1, 2, \ldots, N\}$, and let I, J, and K be a subsets of $\mathcal{V}$, then we can define the causation entropy as the following:

Definition 1 [20]: The causation entropy from the set of processes J to the set of processes I conditioning on the set of processes K is defined as

$$C_{J\to I|K} = H(X_{t+1}^{(I)}|X_t^{(K)}) - H(X_{t+1}^{(I)}|X_t^{(K)}, X_t^{(J)}). \tag{17}$$

The Causation entropy is a natural generalization of transfer entropy from measuring pairwise causal relationships to network relationships of many variables. In particular, we can list the main properties for the causation entropy, noting that if $J = \{j\}$ and $I = \{i\}$, we simplify the notation as $C_{j\to i|K}$:

- If $j \in K$, then the causation entropy $C_{j\to i|K} = 0$, as j does not carry extra information (compared to that of K).
- If $K = \{i\}$, then the causation entropy recovers the transfer entropy $C_{j\to i|i} = T_{j\to i}$ which is given by $T_{j\to i} = H(X_{t+1}^{(i)}|X_t^{(i)}) - H(X_{t+1}^{(i)}|X_t^{(i)}, X_t^{(j)})$..

In [20], we introduced the principle of optimal Causation Entropy (oCSE) in a network of N processes to find the minimum subset that maximizes the causation entropy. This minimal subset can be seen as the dominant subset of a network of N processes, and they rule the underlying dynamic of the network. and in the same principle, we are looking for the dominant terms of the basis function Φ on the system dynamic $\boldsymbol{f}$. See (main text Fig.1) for visualization of the reformulation of Lorenz system in a network of processes.

3 Entropic Regression

In our main text we discussed the Entropic Regression method and provided its Algorithm (main text Algorithm 1). In this section we discuss the tolerance estimation and its effect on the performance of ER.

In our previous work [20] we introduced a standard shuffle test, with a "confidence" parameter $\alpha \in [0, 1]$ for tolerance estimation. The shuffle test requires randomly shuffling of one of the time series n_s times, to build a test statistic. In particular, for the i-th random shuffle, a random permutation $\pi^{(i)} : [T] \to [T]$ is generated to shuffle one of the time series, say, (y_t), which produces a new time series $(\tilde{y}_t^{(i)})$ where $\tilde{y}_t^{(i)} = y_{\pi^{(i)}(t)}$; (x_t) is kept the same. Then, we estimate the mutual information $I(X; \tilde{Y}^{(i)})$ using the (partially) permuted time series $(x_t, \tilde{y}_t^{(i)})$, for each $i = 1, \ldots, n_s$. For given α, we then compute a threshold value $I_\alpha(X; Y)$ as the α-percentile from the values of $I(X; \tilde{Y}^{(i)})$. If $I(X; Y) > I_\alpha(X; Y)$, we determine X and Y as dependent; otherwise independent.

We showed in [20], the robustness of shuffling test for optimal causation entropy calculations specially in complex dynamics, although it is computationally expensive. For more efficient computations complexity, we considered two different approach for tolerance estimation with the confidence parameter α; the Dynamic, and the Static approaches.

In the **Dynamic** tolerance estimation, we start the forward step procedure (See main text Algorithm 1) with initial tolerance $tol = 0$, and we update the tolerance value at the end of the forward step procedure by the shuffle test shown in Algorithm 1 below, given the confidence parameter α, and the current position with the maximum mutual information ϕ, and the conditioning set of strong basis selected so far Φ_K.

The Static approach is more simple and it set to be our default approach since it shows high efficiency in the standard problems such as SID of Lorenz system, while the dynamic approach is more robust for SID of the large complex networks. In the static approach we estimate the tolerance before starting the forward step procedure, and perform no updates during the forward step shown in our main text. That is why we called the approaches dynamic and static.

In **Static** approach, we consider a selected small set of basis functions as a base of estimation, and we will refer to it as the initial conditioning set (ICS) in the following discussion. The ICS is not necessarily to be completely accurate, it just helps to understand the information flow withing the system, and in most cases, any non-relevant basis function included in the ICS will be eliminated during the backward step of ER. In our implementation we considered the linear terms (the x, y, z terms in Lorenz system example) to be our ICS since they are the core of information for all other basis, then, we follow the estimation procedure shown in Algorithm 2, noting that Φ_K is our ICS, and the function π^i represent no shuffling with $i = 0$.

Algorithm 1 Shuffle Test

procedure SHUFFLE TEST($\mathbf{f}, \phi, \mathbf{\Phi_K}, \alpha, \mathbf{n_s}$)
  $i = 1$, $I = \emptyset$
  while $i \leq n_s$ **do**
    $I \leftarrow C_{\phi \rightarrow \mathbf{f}_{\pi^i} | \mathbf{\Phi_K}(\mathbf{\Phi_K^+ f})}$,
    $i := i + 1$,
  return I
  $\mathcal{I} \leftarrow I$ s.t. $\mathcal{I}_j \leq \mathcal{I}_{j+1}$, $j = 1, \ldots, n_s - 1$
  $tol = \mathcal{I}_k$, where $k = \lceil \alpha n_s \rceil$.
return *tol*

The static method of estimating the tolerance, and after initializing the ICS, suggests to accept any basis function that have a small fraction $(1 - \alpha)$ of the maximum possible information from all other basis, and adding the accepted basis function to the conditioning set. This approach does not require high shuffling number, the suggested reasonable value for n_s in the dynamic approach is $n_s = 1000$, while, based in our numerical observations for different systems, $n_s \leq 100$ was quite enough in the static approach.

Algorithm 2 Static Tolerance

procedure SHUFFLE TEST($\mathbf{f}, \mathbf{\Phi}, \mathbf{\Phi_K}, \alpha, \mathbf{n_s}$)
  $i = 0$, $I = \emptyset$
  while $i \leq n_s$ **do** for all basis $\Phi_j \in \Phi$,$j \notin K$
    $I \leftarrow C_{\Phi_j \rightarrow \mathbf{f}_{\pi^i} | \mathbf{\Phi_K}(\mathbf{\Phi_K^+ f})}$,
    $i := i + 1$,
  return I
  $tol = (1 - \alpha) \max(I)$.
return *tol*

Our method considers the system in SID process as a black-box, without assuming the availability for any prior information about the system, and our numerical results shows robustness of the method under this assumption. Practically, prior informations about the systems are available, and that can highly improve the computations efficiency by reasonably setting the expansion order and at least one ICS. If such reasonable settings are not available, ER performs at the same level of accuracy under our black-box assumption, but with more expensive computations.

4 Additional Numerical Results

To demonstrate the utility of ER for nonlinear system identification under noisy observations, we compare its performance against existing methods including the standard least squares (LS), orthogonal least squares (OLS), Lasso, and compressed sensing (CS). The details of the existing approaches are described in the Methods Section. The examples we consider represent different types of systems and scenarios, including both ODEs and PDEs, differential and difference equations, and network-coupled dynamics. In addition, we consider different noise models and especially the presence of outliers in order to evaluate the robustness of the respective methods.

For each example system, we sample the state of each variable at a uniform rate of Δt to obtain a multivariate time series $\{z_k(t_i)\}_{k=1,\ldots,N;i=1,\ldots,\ell}$; then we add noise to each data point and obtain the observed time series denoted by $\{\hat{z}_k(t_i)\}$, where

$$\hat{z}_k(t_i) = z_k(t_i) + \eta_{ki}, \tag{18}$$

with η_{ki} represents noise.

4.1 Double Well Potential

In analogy to the example in our main text, we consider the equation

$$f(x) = x^4 - x^2. \tag{19}$$

and we sample 61 equally spaced measurements for $x \in [-1.2, 1.2]$, and we construct Φ using the $10^t h$ order polynomial expansion with $K = 11$ is the number of candidate functions. Then, we consider a single fixed value corrected measurement to be $f(0.5) = 0.5$.

In this example, we see that the true solution will have a residual δ equal to outliers deviation from its true position,

$$\delta = \sqrt{(f(0.5) - 0.5)^2} = 0.6875 \tag{20}$$

Fig. 5 shows the result for LS. The LS with its BLUE property (Best Linear Unbiased Estimator), succeed to minimize the residual to have better fitting residual than the true solution, but it is clear that the residual value does not reflect reliable solution. Practically, when the true solution gives a fitting residual δ, then any other solution deviate in its residual from δ will have a reduction in the solution accuracy, no matter the direction of deviation from δ. In Fig. 6 we see the result of OLS. We see that the results with best residual of OLS is almost identical to LS result. Here it worth to say a detailed review for the 1000 OLS solutions under different threshold showed us a small interval that gives solutions closer in structure to the true solution more than the minimum residual solution shown, which if treated with suitable trade-off strategy can give a better solution.

Fig. 7 shows the result for CS, where it failed to find any feasible solution for all values of $\epsilon < \delta$. Such outliers makes it hard to find a parameter vector $\boldsymbol{a}$ that can fit the data including the outliers point, and even with considering high resolution for ϵ span, so, CS as discussed before tends to select the solution with minimum $\|\boldsymbol{a}\|_1$ within the best feasible residuals. CS solution simulation for different outliers values is provided on our YouTube channel here. Fig. 8 shows the result for LASSO, and it shows the sparse solution with wrong structure of LASSO. We considered the bounds of λ to be $\lambda \in \left[\|\Phi\Phi^\dagger \boldsymbol{f} - \boldsymbol{f}\|, \|\boldsymbol{f}\|\right]$, where $\lambda = \|\Phi\Phi^\dagger \boldsymbol{f} - \boldsymbol{f}\|$ is the penalty on the solution with all entries are non-sparse and $\lambda = \|\boldsymbol{f}\|$ is the penalty on the solution with all entries are sparse.

Fig. 9 shows the accurate structure found by ER. Even with slight difference in the parameters magnitude, we see how ER recovers the true basis functions. The residual of ER was 0.865, which is higher more than most other methods, but the ER focuses on the information flow between the basis and dynamic and not the residual of solution magnitudes.

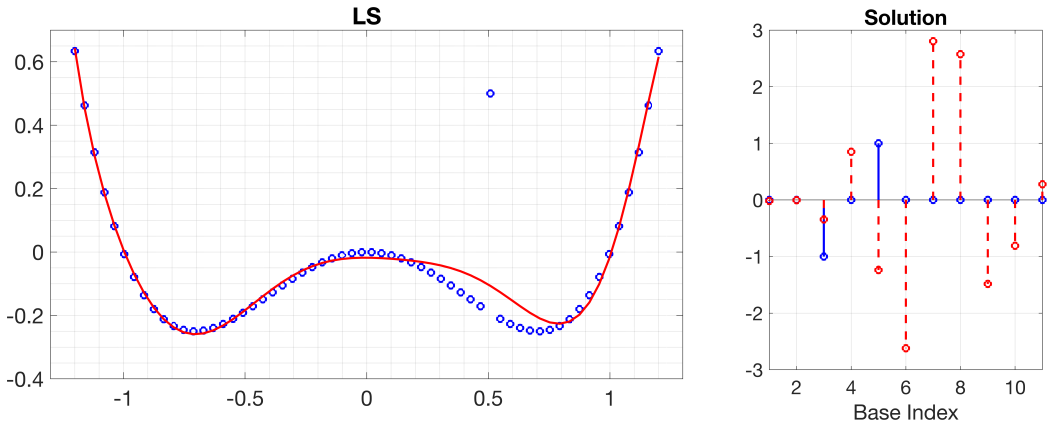


Figure 5: The LS solution for the data given by Eq. 19. This result shows how the LS invest in all available parameters to reach the best possible fitting. In fact, the residual of the least square solution was lower than the residual of the true solution, $0.64 = \|\Phi\Phi^{\dagger}\boldsymbol{f} - \boldsymbol{f}\| < \|\Phi\boldsymbol{a_{true}} - \boldsymbol{f}\| = 0.6875$, and in sparse regression literature, this initiate the need for developing trade off algorithms that considers different measures such as $\|\boldsymbol{a}\|_1$ and $\|\boldsymbol{a}\|_0$.

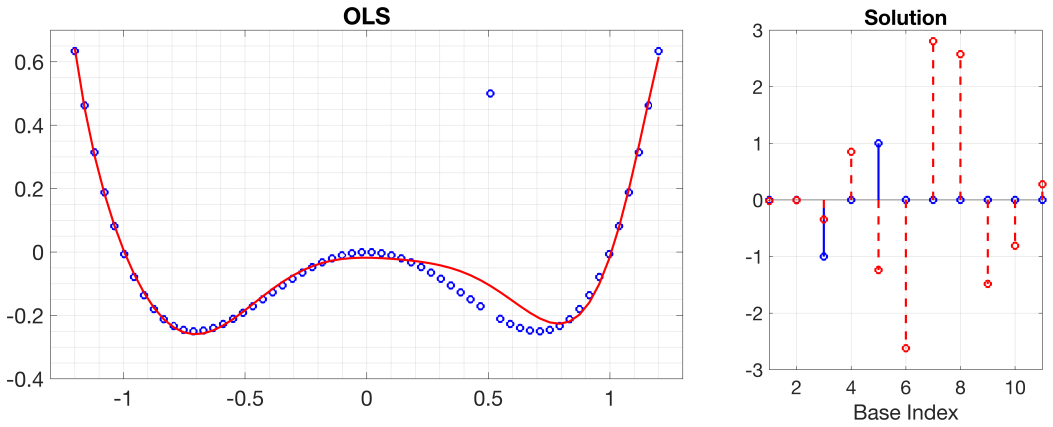


Figure 6: The OLS solution with 1000 log-spaced span for the threshold value $\epsilon \in [10^{-6}, 10^{2}]$. We see that the OLS failed to find solution better than the LS and they are almost identical.

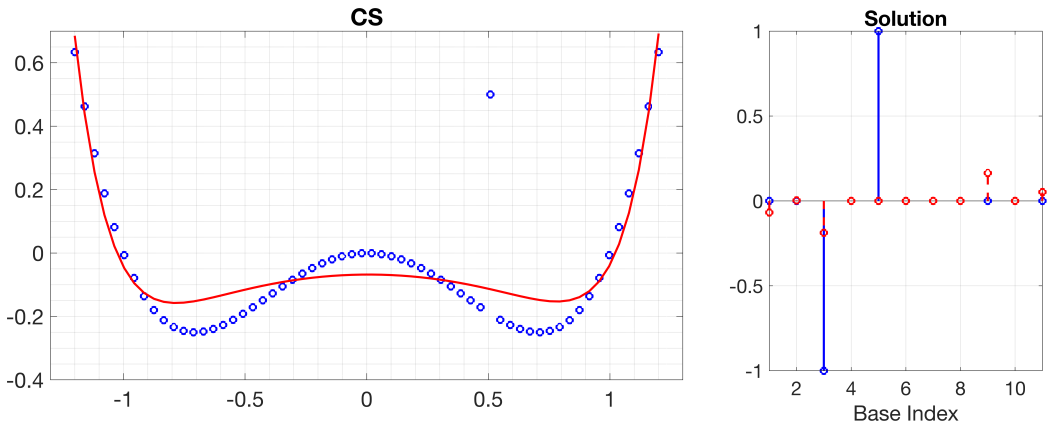


Figure 7: The CS solution, with 1000 log-spaced span for $\epsilon \in [10^{-6}, 10^{2}]$. The solution with minimum residual is shown to the right. As expected, the CVX solver failed to find any feasible solution for all values of $\epsilon < 0.69$, and that was the reason to consider 10^2 as the upper bound of epsilon although it represent a high value for tolerance.

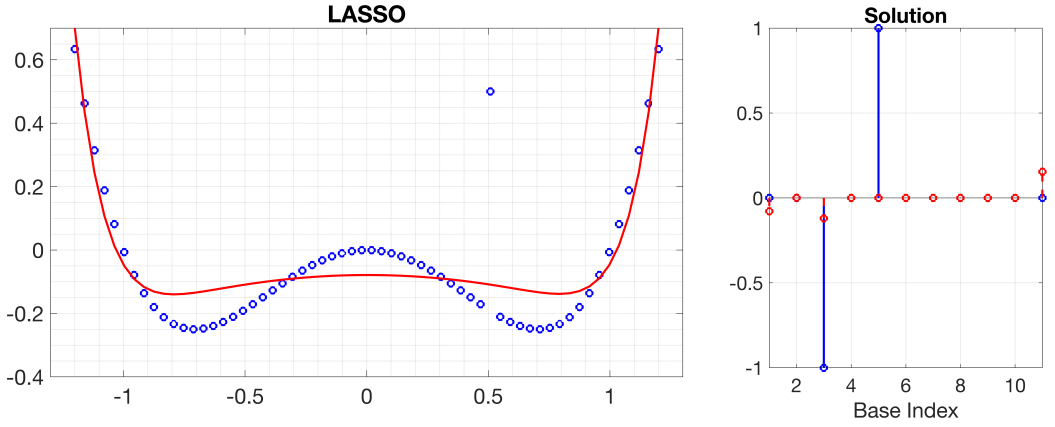


Figure 8: The LASSO solution, with 1000 equally-spaced span for $\lambda \in \left[\|\Phi\Phi^{\dagger}\boldsymbol{f} - \boldsymbol{f}\|, \|\boldsymbol{f}\|\right]$. The solution with minimum residual is shown to the right and it found at $\lambda = 0.818$.

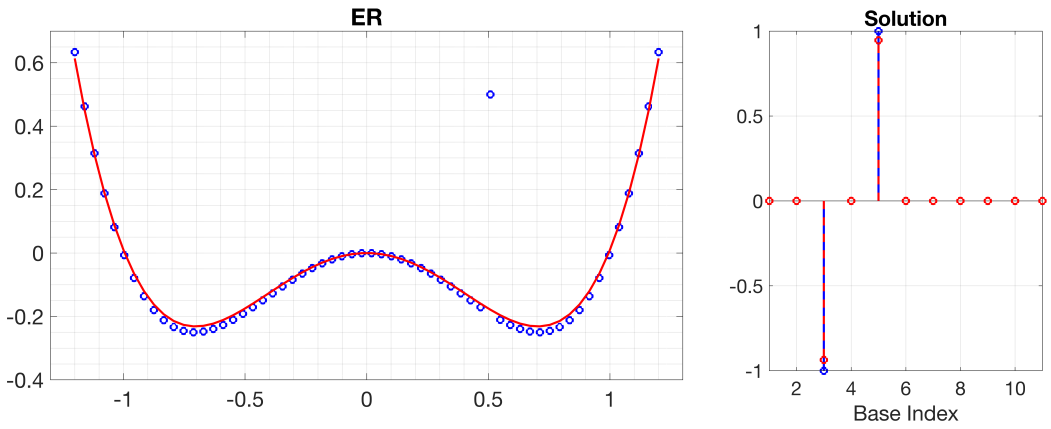


Figure 9: The ER solution. We see that ER recovered the true solution, No trade-off, No-tuning parameter and large span with expensive computations.

4.2 Lorenz system.

Our second detailed example data set was generated by noisy observations from a chaotic Lorenz system. The dynamics of the system is represented by a three-dimensional ODE which is a prototype system as a minimal model for thermal convection, obtained by a low-ordered modal truncation of the Saltzman PDE [16], and for many parameter combinations exhibits chaotic behavior [14]. In our standard notation, we have $\boldsymbol{z} = [z_1, z_2, z_3]^\top$ and

$$\begin{cases} \dot{z}_1 = & F_1(\boldsymbol{z}) = \sigma(z_2 - z_1), \\ \dot{z}_2 = & F_2(\boldsymbol{z}) = z_1(\rho - z_3) - z_2, \\ \dot{z}_3 = & F_3(\boldsymbol{z}) = z_1 z_2 - \beta z_3. \end{cases}$$

We consider a standard polynomial basis:. $[\phi_0, \phi_1, \ldots, \phi_{56}] = [1, z_1, z_2, z_3, z_1^2, z_1 z_2, z_1 z_3, z_2^2, \ldots, z_3^5]$, which contains 56 terms.

In our main text we discussed the example and experiment details for Lorenz system, and we compared in Figs. 2-4 the robust results of ER compared to other methods. Here we extend the results showed in (Main text Fig. 3) to present the results of CS and ER for Lorenz system at different values of p.

Fig. 10 shows the results of CS for different values of p. We see that with $p = 0$, representing no outliers, and small noise $\epsilon_1 = 10^{-5}$, CS recovers the system with high accuracy and the solution can reproduce the dynamic accurately. With only one outliers point out of 1000 measurements used, CS failed to recovering the true structure of the parameters and dynamic produced by the recovered solution diverges after few iterations, and we have similar results with $p = 0.2$ and $p = 0.4$.

Fig. 11 shows the results of ER for different values of p. We see that with $p = 0$, representing no outliers, and small noise $\epsilon_1 = 10^{-5}$, ER recovers the system with high accuracy similar to CS at the same level of noise, while ER continued to perform in the same hight accuracy with the presence of the one outliers point at the same position and magnitude as in CS experiment. With $p = 0.2$ and $p = 0.4$, we see that ER solution continued to reproduce the dynamic, and it normally split from the true dynamic according to the sensitivity to initial conditions. Of course we start from the same initial condition, but the sensitivity here comes from the small difference in parameters magnitude. In (Main text Fig. 4) we showed that ER solution was able to accurately produce the bifurcation diagram of Lorenz system with differences does not exceeds the micro-scale from the true parameters.

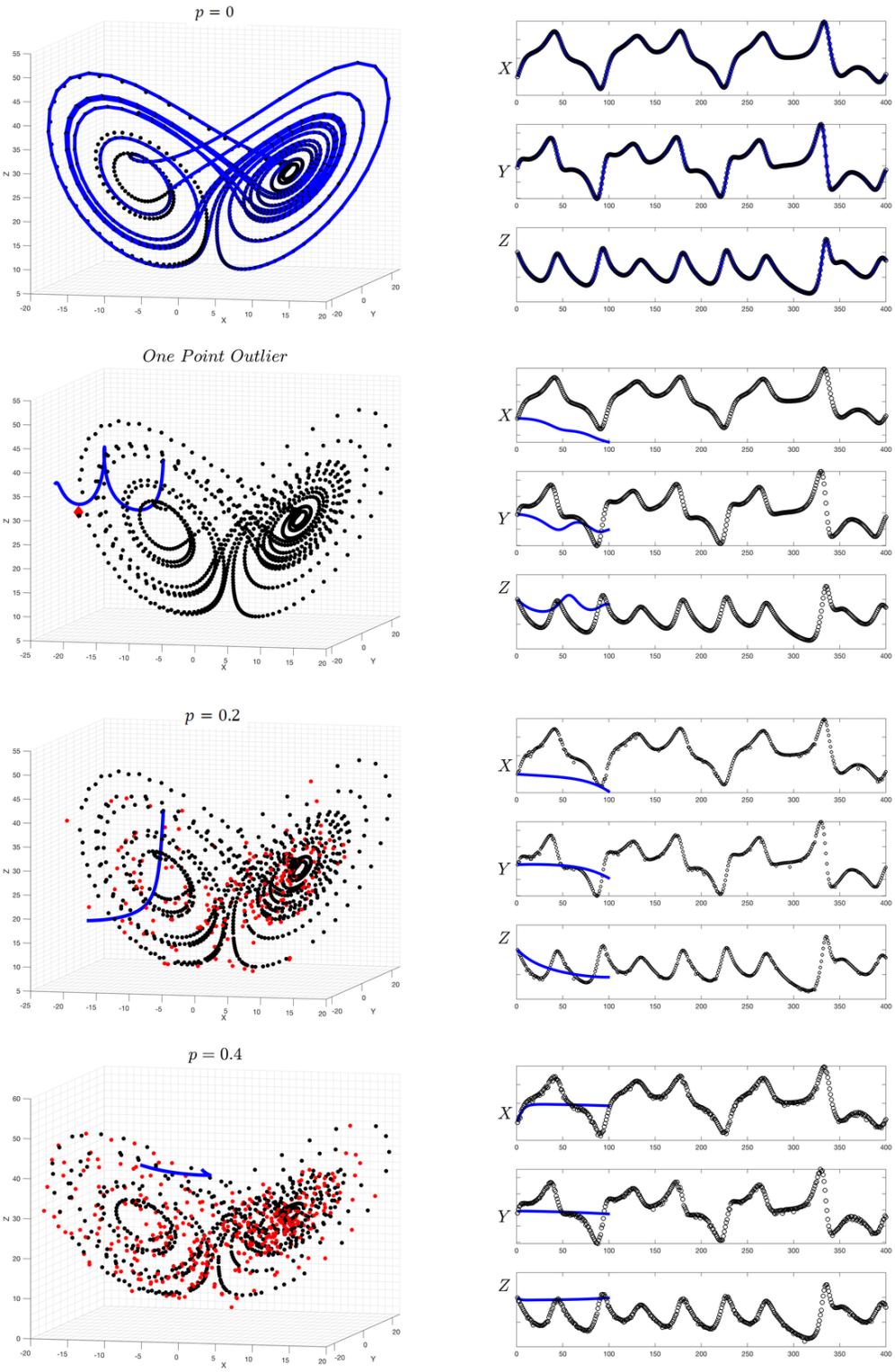


Figure 10: In analogy to (main text Fig.3), CS results for different values of p.

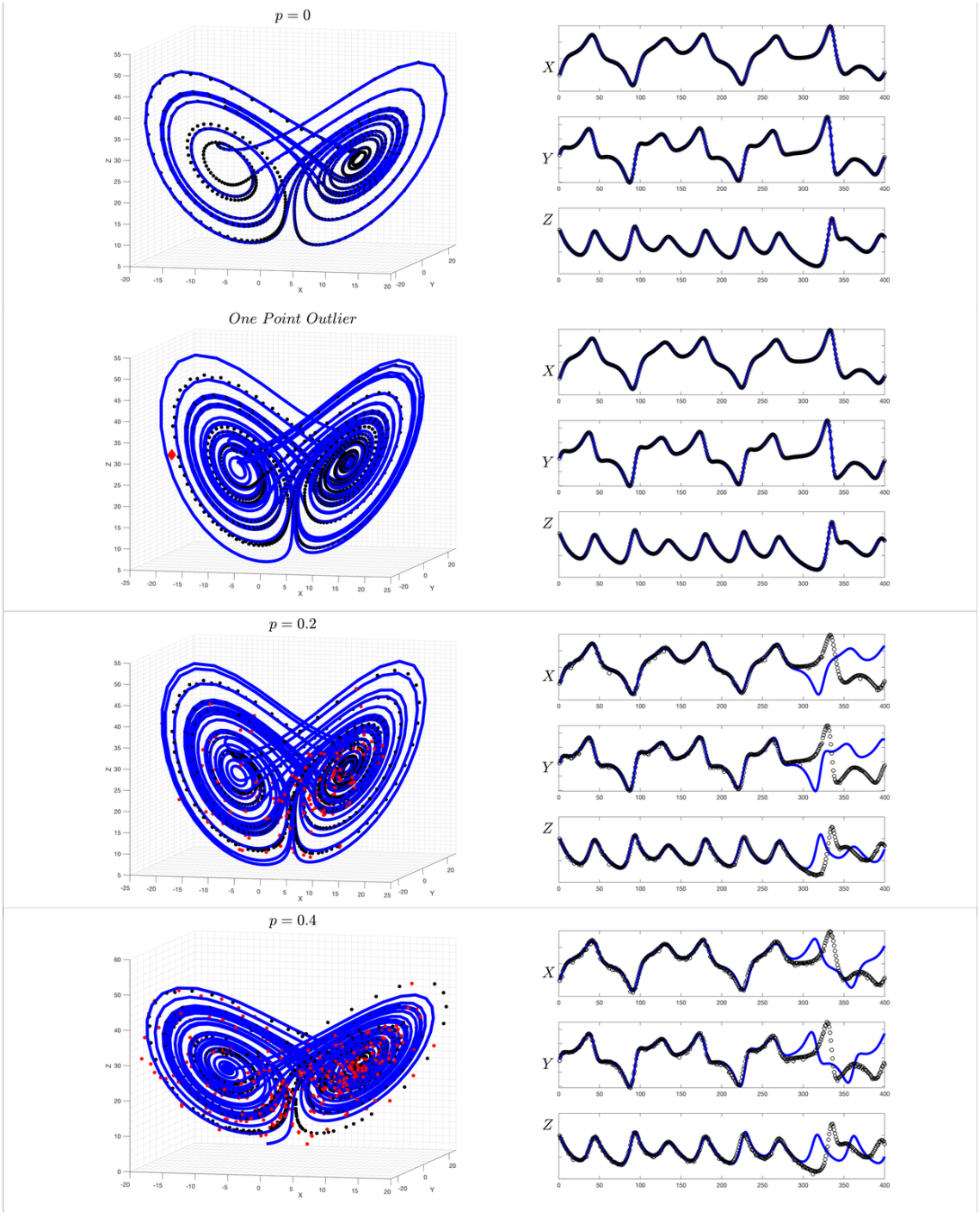


Figure 11: In analogy to (main text Fig.3), ER results for different values of p.

4.3 Network coupled logistic maps.

Our third example is a network of coupled logistic maps which is typical of either coupled map lattices [9], but also cellular automata [10] and more generally the scenario of high dimensional and complex systems that have become the thrust of recent analysis including in the synchronization literature [15, 1, 8]. In this example, we assume that not only the governing dynamics are unknown, but so is the structure of the network that moderates the coupling between individual chaotic elements; both of these must be (simultaneously) identified from observed dynamic data alone. In Fig. 12, we compare results of several system identification methods, including the proposed ER approach. We now offer here a rough description of why this dramatic difference in performance, in the setting particular here of noisy data subject to outliers; a more detailed mathematical analysis will be the subject of our future work.

Consider that each of these other methods we reviewed involves minimizing a functional $J(\mathbf{a})$ of the data a, and that when $\mathbf{a}$ is subject to noise, that the functionals are each continuous with respect to their argument. We assume that the underlying system is,

$$f(x) = ax(1-x), \tag{21}$$

describing the individual elements as Logistic maps, but, the coupled network of N such oscillators is of the form,

$$F(x_i) = f(x_i) + \sum_{j=1}^{N} A_{ij} W_{ij} \left(f(x_j) - f(x_i) \right) \tag{22}$$

where $i, j = 1, ..., N$, A is the adjacency matrix of the coupled network, W_{ij} is the coupling strength between the nodes i and j, and $f(x_i)$ is the image of the point x_i under the logistic map given in Eq.21.

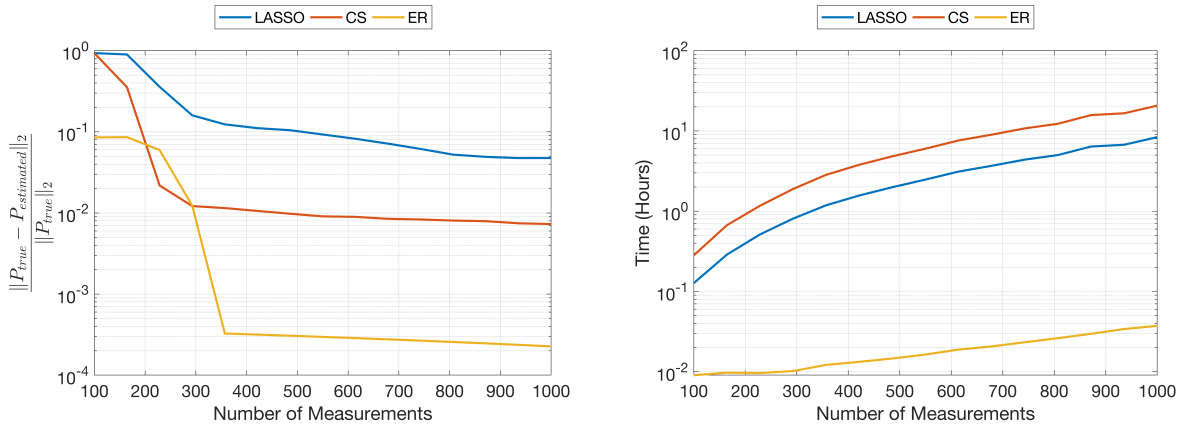


Figure 12: (Left) The relative error in recovered parameters with noise $\epsilon = 10^{-3}$, second order expansion. (Right) The run time to find the parameters. We perform the experiment to find the parameters for one single dimension (one column of the parameters matrix out of the 100 ones) chosen randomly, and results are averaged over only 10 runs. The estimated time required for CS to complete all dimensions and average over 100 runs will be around 20,000 hours (2.3 years).

To present a specific example, let $N = 100$, we construct the adjacency matrix A to have simple coupling such that:

$$1 < D_{ii} \leq 4 \tag{23}$$

where D is the degree matrix of A, and the coupling adjacency matrix A constructed randomly such that the above inequality holds. Then if we consider only the second order expansion we will have 5151 terms in our expansion matrix. LS and OLS will requires then the availability of at least 5152 measurements.

So, we focuses in this example on solving underdetermined system with considering 1000 measurements are available. So, we consider using LASSO, CS and ER in this problem. Fig. 12 shows the relative error and computations complexity for this example. As the computations complexity show, it was hard to perform the excrement to find all the parameters for all oscillators dynamic. So, we perform the experiment to find the parameters for one single dimension chosen randomly, and results are averaged over 10 runs. Time complexity shows that the time required from CS to complete all dimensions and average over 100 runs will be around 20,000 hour (2.3 years), while the time was around 30 hours for ER to complete the 100 dimensions over 100 runs. Fig. 13 shows the sparse structure for the ER solution, and we clearly see the high accuracy of the regression process where we have zero false negative rate and a false positive rate less than 10^{-3}.

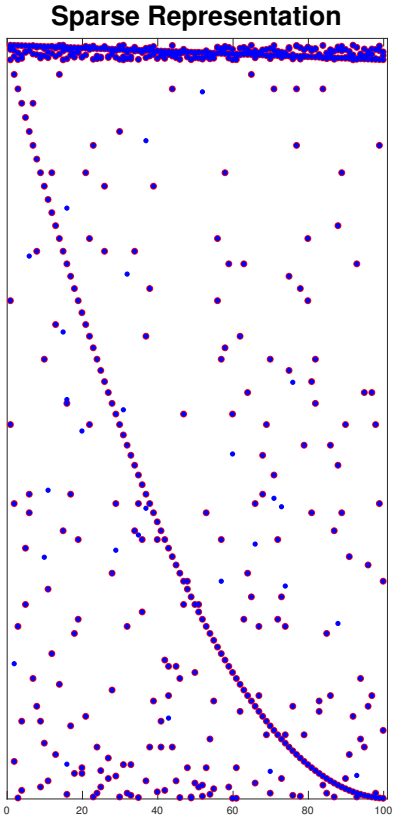


Figure 13: In analogy to (main text Fig. 4), this figure shows the sparse solution of ER. We see that we have few false positive, the false positive 28 out of 515,100 (0.5 million) parameters. And there is zero false negative.

4.4 Kuramoto-Sivashinsky equations.

To further demonstrate the power of ER, we consider a nonlinear PDE, namely the Kuramoto-Sivashinsky (KS) equation [11, 12, 19, 7, 13], which arises as a description of flame front flutter of gas burning in a cylindrically symmetric burner. It has become a popular example of a PDE that exhibits chaotic behavior,

in particular spatiotemporal chaos [2, 6]. We will consider Kuramoto-Sivashinsky system

$$u_t = -\nu u_{xxxx} - u_{xx} + 2uu_x, \qquad (t,x) \in [0,\infty) \times (0,L) \tag{24}$$

in periodic domain, $u(t,x) = u(t,x+L)$, and we restrict our solution to the subspace of odd solutions $u(t,-x) = -u(t,x)$. The viscosity parameter ν controls the suppression of solutions with fast spatial variations, and we have chaotic solution when $\nu = 0.029910$, [2].

Since a PDE corresponds to an infinite dimensional dynamical system, in practice we focus on an approximate finite-dimensional model of the system, for example, by Galerkin-projection onto basis functions as infinitely many ODE's in the corresponding Banach space.

In our main text we discussed the example and experiment details for KSE system, and we showed in Figs. 5-6 the results of ER. Here we extend the results to present the comparison between different solvers results.

Fig. 14 shows the results for the different solvers for KSE example, KSE showed high sloppiness in the parameters where the **Sloppy Parameters** can be a challenging problem for different parameters estimation methods [4, 22]. In CVX optimization, which is the case with CS and LASSO, as of most of other optimization methods, the search mechanism for the optimal solution depends on finding the good search directions in the search space based on the response of the objective function to the change of parameters, which become challenging and very expensive task with the presence of sloppy parameters.

From (Main text Eq. 15), we see that the KSE parameters can grow to a very high values (of order 10^5) with number of modes $N_m > 20$, while in the same time we have parameter with values less than 1 at the low index modes. Then, the OLS (which depends mainly on the response of fitting residual) showed high tendency to oversparse the sloppy parameters, and according to there high magnitude, oversparsing the higher magnitude parameters results with relative error in the solution very close to 1. The LS does not care to the parameters magnitude, it only aims to reduce the residual and it invest in all possible directions for this purpose, and here we see the other side of the sloppiness effect, where we see in Fig. 14 that the LS solution "relative" error grows to a very high values according to investing large energy in sloppy locations on the parameters matrix.

With low number measurements, ER had fuzzy information detection process and was not able to detect the structure accurately, and once we have enough measurements to accurately detect the information flow, ER was able to recover the parameters matrix accurately.

5 ER codes in Matlab and User Guide

A full Matlab code for the entropic regression solver and other core functions such as mutual information estimator, conditional mutual information estimator, and data generator are available at GitHub in addition to many other tools functions. The code provided with full documentations and explanations where we worked to make it simple to follow up and use.

In this section we introduce a simple guide to show the simple process of using the entropic regression algorithm, and to show a sample **real time** results for the entropic regression for the basic well known systems; Lorenz system, Rossler system, and Vander Pol system. Note that the results shown generated through Matlab publishing tool, meaning that it is a real time run with the same settings and options for all the results. To download the code and related documentations see: GitHub.

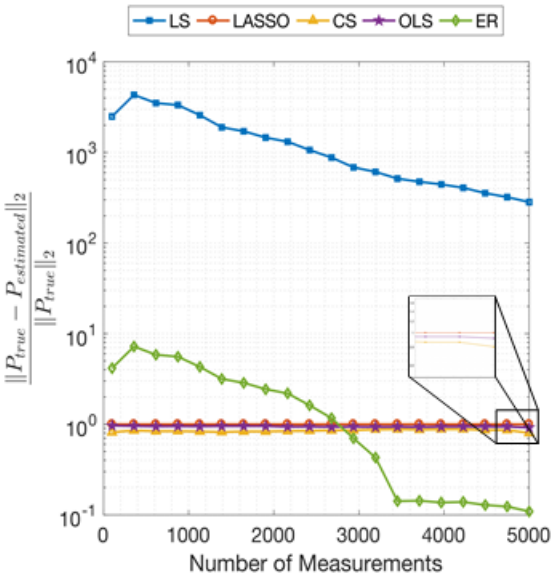


Figure 14: The relative error in estimated parameters with $N_m = 25$ and $\epsilon_1 = 0.001$. LASSO, CS and OLS, all of them always over sparse, they only give very few non-zero parameters, which result with their relative error to be almost 1 with very small differences as shown in the zoom region.

Entropic Regression Fitting

Given Φ and f, *(erfit.m)* solve the problem:

$$f = \Phi x$$

for x. We provide the matlabe code for entropic regression fitting which can be found at GitHub and FileExchange in addition to a sample data generator *(dataGen.m)*. The default settings for the (erfit) function is prepared to deal with the regression as a black-box without assuming any prior information about the system. For some applications, prior information may be available, and for such cases, we provide the user with different options (see erfit help documentation) that can highly reduce the computations complexity.

In this document, we introduce different examples to get the user familiar with using the erfit. Please, refer to *dataGen.m* for more information about data generation options.

As a summary for (dataGen.m) function, consider the example call:

```
[Phi,f] = dataGen('Lorenz','SampleSize',500);
```

which will generate 500 sample points of the Lorenz system. Now, we have the dynamic of Lorenz system $\dot{x}, \dot{y}$ and $\dot{z}$ stored in the columns of the variable (f), and the fifth order polynomial expansion is the columns of matrix Phi. As mentioned above, and assuming No prior information about the system are available, we simply call the erfit function by:

```
x = erfit(Phi,f);
```

An extra information are available for the data generation and the entropic regresiion process through the optional output (Info) where:

```
[Phi,f,Info] = dataGen('Lorenz','SampleSize',500);
```

Info will provide extra information such as the step size used in integration, the initial condition, the noise, ..., etc. The syntax

```
[x,erInfo] = erfit(Phi,f);
```

will provide the structure erInfo which has detailed information about the regression process.

Standard Results with Noisy Measurements

In this section we will introduce some results for a well known chaotic systems.

```
rng('default'); %for repeatability
```

Logistic Map and Basis Construction

The logistic map is a polynomial mapping (recurrence relation), often cited as an archetypal example of how complex, chaotic behavior can arise from very simple non-linear dynamical equations. And it can be given by:

$$x_{n+1} = f(x_n) = ax_n(1 - x_n)$$

where x_n is a number between zero and one that represents the ratio of existing population to the maximum possible population, and $f : [0\,1] \to [0\,1]$, with the parameter $a = 4$ is commonly used and known to produce chaotic behavior. In this example, we will construct the basis expansion and problem construction in details without calling the DataGen function.

```
func = @(x) 4*x.*(1-x); %logistic map function
x = rand; %Initial condition of logistic map

%Now, we carry on 500 iteration of logistic map
for i=1:500, x = cat(1,x , func(x(end))); end
```

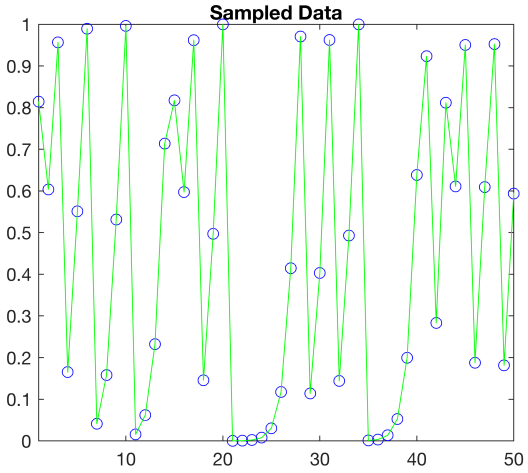
It can be seen that, the measuremnt x_n gives the observation $x_{n+1} = f(x_n)$, then our observations vector is:

```
f = x(2:end); x(end) = [];
```

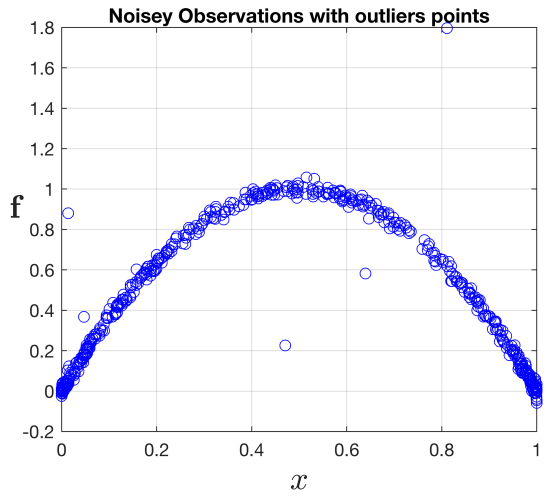
we ignor the last measurement since it is not assigned to observation. Adding gaussian noise $R \sim \mathcal{N}(0, 0.02)$ can be done by:

```
f = f + 0.02*randn(size(f));
f(randperm(length(f),5)) = 2*randn(5,1);

figure('Units','centimeters','Position',[25 25 12 10]);
plot(x,'ob'); hold on
plot(x,'-g');
title('Sampled Data')
xlim([1 50]);%show only 50 iterations for clear view
```



```
figure('Units','centimeters','Position',[25 25 12 10]);
plot(x,f,'ob'); grid on;
xlabel('$x$','Interpreter','latex','FontSize',18)
ylabel('$\mathbf{f}$','Interpreter','latex','FontSize',18,'Rotation',0)
title('Noisey Observations with outliers points')
```



The basis matrix with 5^{th} order polynomial expansion can be constructed as:

```
Phi = [ones(size(x)), x, x.^2, x.^3, x.^4, x.^5];
Sol = [      0      , 4,  -4 ,  0  ,  0  ,  0  ]';
```

where (Sol) stores the true solution of the system. Now, we call the erfit (ER solver) with its default setting to find the system parameters:

```
[x,Inff] = erfit(Phi,f);
disp(table(x, Sol,...
     'VariableNames',{'ER_Solution','True_Solution'}));
```

ER_Solution	True_Solution
0	0
3.9372	4
-3.9405	-4
0	0
0	0
0	0

Lorenz System

The well known Lorenz system given by:

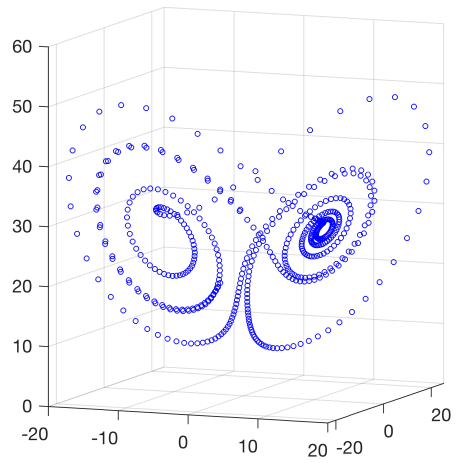
$$\dot{x} = \sigma(y - x)$$
$$\dot{y} = x(\rho - z) - y$$
$$\dot{z} = xy - \beta z$$

a commonly used values of the parameters are $\sigma = 10, \beta = 8/3$ and $\rho = 28$. First, we generate 500 noisy measurements.

```
rng('default');
[Phi,f, Info] = dataGen('Lorenz','SampleSize',500,...
                                 'NoiseLevel',0.1);
```

please refer to our main text for ϵ_1 and ϵ_2 noise and corruption standard deviation.

```
%show the sampled measurements
figure('Units','centimeters','Position',[25 25 10 10]);
scatter3(Info.X(:,1),Info.X(:,2),Info.X(:,3),8,'b')
view([22.5 6.8])
```



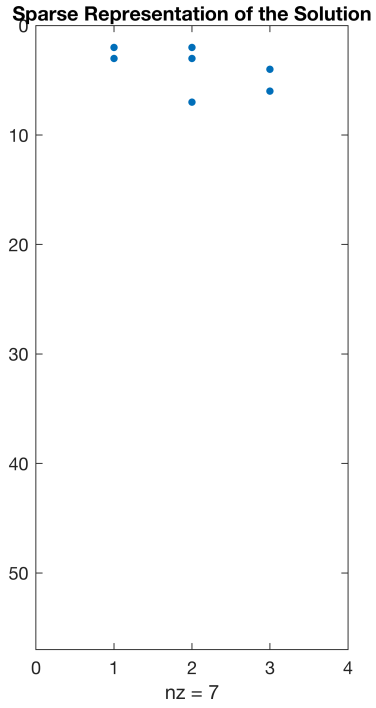
Then, solving the system with entropic regression gives the result:

```
x = erfit(Phi,f);
disp(table(x(1:10,:), Info.P(1:10,:),...
     'VariableNames',{'ER_Solution','True_Solution'}));
```

ER_Solution			True_Solution		
0	0	0	0	0	0
-9.9991	28	0	-10	28	0
9.9987	-1.001	0	10	-1	0
0	0	-2.6656	0	0	-2.6667
0	0	0	0	0	0
0	0	1.0002	0	0	1

```
    0    -0.99999          0      0         -1          0
    0           0          0      0          0          0
    0           0          0      0          0          0
    0           0          0      0          0          0
```

```
figure('Units','centimeters','Position',[0 0 10 15]);
spy(x); pbaspect([1 2 1]);
title('Sparse Representation of the Solution')
```



Corrupted Measurements

Now, we add assume that some of our measurements are corrupted. Please refere to our main text for the discussion of corrupted measurements.

```
rng('default');
[Phi,f, Info] = dataGen('Lorenz','SampleSize',1000,...
                                 'CorruptionStd',  5,...
                                 'CorruptedProb',  0.5);
```

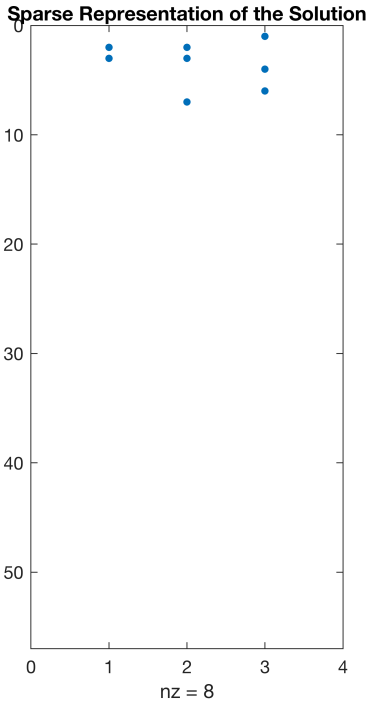
where CorruptionStd is the standard deviation of the corruption and CorruptedProb is the probability that the measurement is corrupted. The above syntax produces 1000 measurements with 50% of them corrupted with high noise. The ER results for such highly corrupted measurements are:

```
x = erfit(Phi,f);
disp(table(x(1:10,:), Info.P(1:10,:),...
```

```
    'VariableNames',{'ER_Solution','True_Solution'}));
```

ER_Solution			True_Solution		
0	0	-0.40753	0	0	0
-9.9726	28.147	0	-10	28	0
9.9569	-1.0356	0	10	-1	0
0	0	-2.6449	0	0	-2.6667
0	0	0	0	0	0
0	0	0.99916	0	0	1
0	-1.0033	0	0	-1	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

```
figure('Units','centimeters','Position',[0 0 10 15]);
spy(x); pbaspect([1 2 1]);
title('Sparse Representation of the Solution')
```



We see that the ER solution has 1 false negative location out of 168 total parameters, and it was the constant term in the third dimention parameters.

Systems have different sensitivity to the corruption magnitude and corruption probability, which should be considered in preparing the data for the regression process in real-world problems.

Entropic Regression

Entropic regression parameters estimator.

Syntax

```
x = erfit(Phi, f);
x = erfit(Phi, f, options);
[x,Info] = erfit(Phi, f, options);
[sol,Info, Mask] = erfit(Phi, f, options);
```

Description

- *x = erfit(Phi, f)* : Given sampled data for basis functions Phi $\in \mathbb{R}^{l \times K}$ and sampled observations, $f \in \mathbb{R}^{l \times d}$, erfit finds the sparse solution $x \in \mathbb{R}^{K \times d}$ that containes the true governing parametrs of the dynamic f.
- *x = erfit(Phi, f, options)* : erfit accept options structure *'options'* that controls the computations created by *eroptset* function.
- *[x,Info] = erfit(Phi, f, options)* : *erfit* provides output structure *Info* that have information about regression process.
- *[x,Info, Mask] = erfit(Phi, f, options)* : *erfit* provides a logical matrix *Mask* where $Mask_{i,j} = 1$ if $x_{i,j} \neq 0$, and $Mask_{i,j} = 0$ otherwise.

Examples

Please refer to our main text for the construction of the Phi and f matrices. The following is a simplified data construction and function call for *erfit*.

```
[Phi, f] = dataGen('Lorenz');
options = eroptset('sbsMethod','dynamic', 'alpha', 0.95);
x = erfit(Phi, f, options);
```

Version

This function is a part of Entropic Regression Software Package (erfit), version 1.0. To report bugs, comments and suggestions, we appreciate your feedback: Abd AlRahman R. AlMomani, almomaa@clarkson.edu.

Function Body

```
function [sol,Info, Mask] = erfit(Phi, f, options)

if nargin < 3 %If no options provided, load defaults
    options = eroptset('pDim',size(f,2));
end

%extract system matrcies dimensions
[stat.M,stat.D] = size(Phi); stat.dim = size(f,2);
```

```
%Solution placeholder
sol = zeros(stat.D,stat.dim);

fwstat = stat; %Initialize forward stat information.

for i=1:stat.dim
    stat.dimIX = i;
    %Strong Basis Selection (Forward Step).
    strongIX = sbs(Phi, f(:,i), stat, options);

    % Update information
    Info(i).fwstat = fwstat;
    Info(i).strIX  = strongIX;


    %Weak Basis Removal (Backward Step)
    optimalIX = wbr( Phi(:,strongIX),f(:,i),Info(i).fwstat, options );
    optimalIX = strongIX(optimalIX);

    % Update information
    Info(i).bwstat = bwstat;
    Info(i).optIX  = optimalIX;



    %If not detected through entropic regression,
    %Include the constant term for influence test.
    if ~ismember(1,optimalIX), optimalIX = cat(2,1,optimalIX); end

    %Given the optimal basis, find least squares solution
    sol(optimalIX,i) = nls( Phi(:,optimalIX), f(:,i) );

    % Update information
    Info(i).mask = logical(sol(:,i));

end
% Find the solution logical mask required
if nargout == 3, Mask = cat(2,Info.mask); end
```

References

[1] C. Anteneodo, A. M. Batista, and R. L. Viana. Synchronization threshold in coupled logistic map lattices. *Physica D: Nonlinear Phenomena*, 2006.

[2] F. Christiansen, P. Cvitanovi??, and V. Putkaradze. Spatiotemporal chaos in terms of unstable recurrent patterns. *Nonlinearity*, 1997.

[3] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory*. 2005.

[4] Bryan C. Daniels, Yan Jiun Chen, James P. Sethna, Ryan N. Gutenkunst, and Christopher R. Myers. Sloppiness, robustness, and evolvability in systems biology, 2008.

[5] C W J Granger. Investigating Causal Relations by Econometric Models and Cross-spectral Methods. *Econometrica*, 37(3):424–438, 1969.

[6] P. C. Hohenberg and Boris I. Shraiman. Chaotic behavior of an extended system. *Physica D: Nonlinear Phenomena*, 1989.

[7] James M. Hyman and Basil Nicolaenko. The Kuramoto-Sivashinsky equation: A bridge between PDE'S and dynamical systems. *Physica D: Nonlinear Phenomena*, 1986.

[8] Sarika Jalan and R. E. Amritkar. Synchronized clusters in coupled map networks. *Proceedings of the Indian National Science Academy Part A - Physical Sciences*, 2005.

[9] Kunihiko Kaneko. Overview of coupled map lattices. *Chaos (Woodbury, N.Y.)*, 1992.

[10] F. Kaspar and H. G. Schuster. Easily calculable measure for the complexity of spatiotemporal patterns. *Physical Review A*, 1987.

[11] Y. Kuramoto and T. Tsuzuki. Persistent Propagation of Concentration Waves in Dissipative Media Far from Thermal Equilibrium. *Progress of Theoretical Physics*, 1976.

[12] Yoshiki Kuramoto. Diffusion-Induced Chaos in Reaction Systems. *Progress of Theoretical Physics Supplement*, 1978.

[13] Yueheng Lan and Predrag Cvitanović. Unstable recurrent patterns in Kuramoto-Sivashinsky dynamics. *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics*, 2008.

[14] Edward N. Lorenz. Deterministic Nonperiodic Flow. *Journal of the Atmospheric Sciences*, 1963.

[15] C. Masoller, Hugo L.D.de S. Cavalcante, and J. R.Rios Leite. Delayed coupling of logistic maps. *Physical Review E - Statistical Physics, Plasmas, Fluids, and Related Interdisciplinary Topics*, 2001.

[16] Barry Saltzman. Finite Amplitude Free Convection as an Initial Value ProblemI. *Journal of the Atmospheric Sciences*, 1962.

[17] T Schreiber. Measuring information transfer. *Physical review letters*, 85(2):461–4, 2000.

[18] Claude E Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(July 1928):379–423, 1948.

[19] G. I. Sivashinsky. Nonlinear analysis of hydrodynamic instability in laminar flames-I. Derivation of basic equations. *Acta Astronautica*, 1977.

[20] J. Sun, D. Taylor, and E. Bollt. Causal network inference by optimal causation entropy. *SIAM Journal on Applied Dynamical Systems*, 14(1):73–106, 2015.

[21] Jie Sun and Erik M. Bollt. Causation entropy identifies indirect influences, dominance of neighbors and anticipatory couplings. *Physica D: Nonlinear Phenomena*, 267:49–57, 2014.

[22] Andrew White, Malachi Tolman, Howard D. Thames, Hubert Rodney Withers, Kathy A. Mason, and Mark K. Transtrum. The Limitations of Model-Based Experimental Design and Parameter Estimation in Sloppy Systems. *PLoS Computational Biology*, 2016.

Acknowledgements

This work was funded in part by the Simons Foundation Grant No. 318812. We would also like to thank, the Army Research Office (N68164-EG) and the Office of Naval Research (N00014-15-1-2093), and also DARPA.