Statistical Inference for Sparse Reconstruction of Dynamical Systems

Sara Venkatraman

Cornell University, Statistics and Data Science

Joint work with Sumanta Basu and Martin Wells

Statistical problem of interest:

How can we use time series data $x(t_1), ..., x(t_n)$ to learn the form of an unknown differential equation $\mathrm{d}x / \mathrm{d}t = f(x(t))$?

Outline:

1. Example: Lotka-Volterra equations
2. Sparse regression for learning differential equations
3. Leveraging recent work in high-dimensional inference

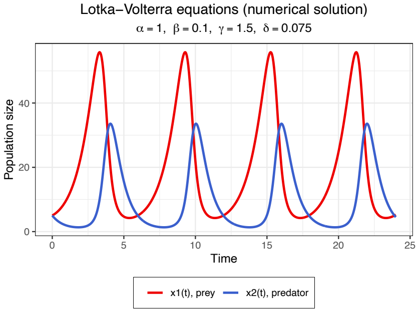
Example from ecology: predator-prey dynamics

The Lotka-Volterra equations describe how the populations of a prey species, $x_1(t)$, and a predator species, $x_2(t)$, evolve in time:

$$\frac{dx_1}{dt} = \alpha x_1 - \beta x_1 x_2, \qquad \frac{dx_2}{dt} = \delta x_1 x_2 - \gamma v$$

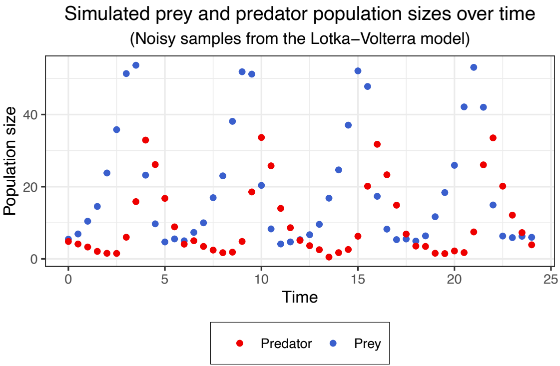
where:

- αx_1 = prey population's rate of increase
- $\beta x_1 x_2$ = prey rate of decrease due to predation
- $\delta x_1 x_2$ = predator rate of increase due to prey availability
- γx_2 = predator rate of decrease



Example from ecology: predator-prey dynamics

Would like to recover the equations for $\mathrm{d}x_1/\,\mathrm{d}t$ and $\mathrm{d}x_2/\,\mathrm{d}t$ from (noisy) temporal population size data:



Then we could numerically solve the learned equations to do simulations, forecasting, etc.

Differential equation learning: problem setup

Suppose the temporal evolution of $\mathbf{x}(t) \in \mathbb{R}^d$ is governed by

$$\frac{\mathrm{d}\mathbf{x}}{\mathrm{d}t} = \mathbf{f}(\mathbf{x}(t)), \text{ for some unknown } \mathbf{f} : \mathbb{R}^d \to \mathbb{R}^d.$$

Given time series data

$$\{\mathbf{x}(t_1),\ \mathbf{x}(t_2),\ \dots,\ \mathbf{x}(t_n)\},$$

how can we learn $\mathbf{f}$ in closed form?

Sparse reconstruction of ODEs

Approach will be based on sparse linear regression (SINDy[1]):

- Assume $\mathbf{f}$ has a sparse representation in some basis, e.g. polynomials of $\mathbf{x}(t)$ components: $x_1(t), ..., x_d(t)$
- Why: Many systems can be written as a small linear combination of $x_1(t), ..., x_d(t)$ (or products of them)
- What's new: Will use new theory for sparse regression to assess the statistical significance of each term in the reconstructed $\mathbf{f}$

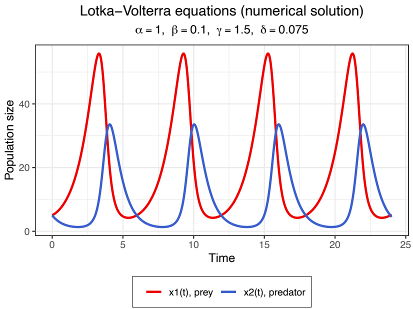
[1] S. Brunton, J. Proctor, J. Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *PNAS*, 2016.

Writing ODEs as linear systems: a 2D example

Lotka-Volterra equations:

$$\frac{\mathrm{d}x_1}{\mathrm{d}t} = \alpha x_1 - \beta x_1 x_2$$

$$\frac{\mathrm{d}x_2}{\mathrm{d}t} = \delta x_1 x_2 - \gamma x_2$$



can be written as:

$$\begin{bmatrix} \frac{\mathrm{d}x_1}{\mathrm{d}t} & \frac{\mathrm{d}x_2}{\mathrm{d}t} \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & x_1 x_2 \end{bmatrix} \begin{bmatrix} \alpha & 0 \\ 0 & -\gamma \\ -\beta & \delta \end{bmatrix}$$

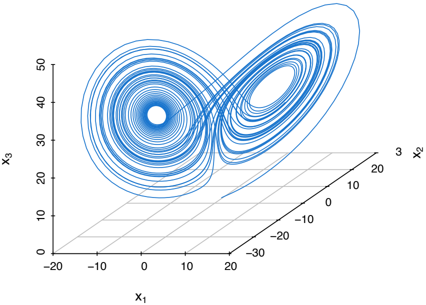
Writing ODEs as linear systems: a 3D example

Lorenz equations:

$$\frac{\mathrm{d}x_1}{\mathrm{d}t} = \sigma(x_2 - x_1)$$

$$\frac{\mathrm{d}x_2}{\mathrm{d}t} = x_1(\rho - x_3) - x_2$$

$$\frac{\mathrm{d}x_3}{\mathrm{d}t} = x_1 x_2 - \beta x_3$$



can be written as:

$$\begin{bmatrix} \frac{\mathrm{d}x_1}{\mathrm{d}t} & \frac{\mathrm{d}x_2}{\mathrm{d}t} & \frac{\mathrm{d}x_3}{\mathrm{d}t} \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & x_3 & x_1x_2 & x_1x_3 \end{bmatrix} \begin{bmatrix} -\sigma & \rho & 0 \\ \sigma & -1 & 0 \\ 0 & 0 & -\beta \\ 0 & 0 & 1 \\ 0 & -1 & 0 \end{bmatrix}$$

Sparse reconstruction of ODEs

Approach to estimating $\mathbf{f} : \mathbb{R}^d \to \mathbb{R}^d$ in $\mathrm{d}\mathbf{x}/\,\mathrm{d}t = \mathbf{f}(\mathbf{x}(t))$:

- Assume each component of $\mathbf{f}$ is a sparse linear combination of $x_1, ..., x_d$ and their polynomials up to degree k
- E.g. if dimension $d = 2$ and degree $k = 2$ we have:

$$\underbrace{\begin{bmatrix} \dot{x}_1 & \dot{x}_2 \end{bmatrix}}_{\dot{\mathbf{x}} = \mathbf{f}} = \underbrace{\begin{bmatrix} 1 & x_1 & x_2 & x_1 x_2 & x_1^2 & x_2^2 \end{bmatrix}}_{\boldsymbol{\Theta}(\mathbf{x})} \underbrace{\begin{bmatrix} \beta_{01} & \beta_{02} \\ \beta_{11} & \beta_{12} \\ \beta_{21} & \beta_{22} \\ \beta_{31} & \beta_{32} \\ \beta_{41} & \beta_{42} \\ \beta_{51} & \beta_{52} \end{bmatrix}}_{\mathbf{B}}$$

- Or in matrix form, $\dot{\mathbf{x}} = \boldsymbol{\Theta}(\mathbf{x})\mathbf{B}$

Remarks about matrix representation of ODEs

$$\underbrace{\begin{bmatrix} \dot{x}_1 & \dot{x}_2 \end{bmatrix}}_{\dot{\mathbf{x}} = \mathbf{f}} = \underbrace{\begin{bmatrix} 1 & x_1 & x_2 & x_1 x_2 & x_1^2 & x_2^2 \end{bmatrix}}_{\boldsymbol{\Theta}(\mathbf{x})} \underbrace{\begin{bmatrix} \beta_{01} & \beta_{02} \\ \beta_{11} & \beta_{12} \\ \beta_{21} & \beta_{22} \\ \beta_{31} & \beta_{32} \\ \beta_{41} & \beta_{42} \\ \beta_{51} & \beta_{52} \end{bmatrix}}_{\mathbf{B}}$$

- Objective: estimate $\mathbf{B}$ from time series data $\mathbf{x}(t_1), \ldots, \mathbf{x}(t_n)$
- $\mathbf{B}$ should be sparse since many ODEs have only a few terms
- $\boldsymbol{\Theta}(\mathbf{x})$ should be large enough to contain true terms in unknown $\mathbf{f}$
- Large $\boldsymbol{\Theta}(\mathbf{x})$ means this could become a high-dimensional problem

A regression problem

Data is assumed to be noisy so our model is:

$$\dot{\mathbf{X}} = \mathbf{\Theta}(\mathbf{X})\mathbf{B} + \varepsilon$$

E.g. for a 2D system and degree-2 polynomials we have:

$$\dot{\mathbf{X}} = \underbrace{\begin{bmatrix} \dot{x}_1(t_1) & \dot{x}_2(t_1) \\ \dots & \dots \\ \dot{x}_1(t_n) & \dot{x}_2(t_n) \end{bmatrix}}_{\substack{\text{matrix of time derivatives} \\ \text{(computed numerically)}}}, \quad \mathbf{B} = \underbrace{\begin{bmatrix} \beta_{01} & \beta_{02} \\ \dots & \dots \\ \beta_{51} & \beta_{52} \end{bmatrix}}_{\substack{\text{sparse matrix of} \\ \text{coeffs. (unknown)}}}, \quad \varepsilon \text{ is i.i.d. } N(0, \sigma^2) \text{ noise,}$$

$$\mathbf{\Theta}(\mathbf{X}) = \underbrace{\begin{bmatrix} 1 & x_1(t_1) & x_2(t_1) & x_1(t_1)x_2(t_1) & x_1^2(t_1) & x_2^2(t_1) \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & x_1(t_n) & x_2(t_n) & x_1(t_n)x_2(t_n) & x_1^2(t_n) & x_2^2(t_n) \end{bmatrix}}_{\text{polynomials of observed data}}$$

Derivative estimation

How to estimate the entries of $\dot{\mathbf{X}} = \begin{bmatrix} \dot{\mathbf{x}}(t_1)^T \\ \dots \\ \dot{\mathbf{x}}(t_n)^T \end{bmatrix}$:

- Finite difference approximations:

$$\dot{\mathbf{x}}(t_k) \approx \frac{\mathbf{x}(t_{k+1}) - \mathbf{x}(t_k)}{t_{k+1} - t_k}, \quad \ddot{\mathbf{x}}(t_k) \approx \frac{\mathbf{x}(t_{k+1}) - 2\mathbf{x}(t_k) + \mathbf{x}(t_{k-1})}{(t_{k+1} - t_k)^2}$$

  but these are sensitive to noise in the $\mathbf{x}(t_k)$'s.

- Polynomial interpolation: approximate $\dot{x}_j(t_k)$ by fitting a polynomial through $\{x_j(t_i)\}_{i=1}^n$ and differentiating it

- Denoising methods: e.g. total variation regularization, spectral filtering, ...

Estimating $\mathrm{d}\mathbf{x}/\mathrm{d}t = \mathbf{f}(\mathbf{x})$ via $\dot{\mathbf{X}} = \mathbf{\Theta}(\mathbf{X})\mathbf{B} + \varepsilon$

Recall that $\dot{\mathbf{X}} = \mathbf{\Theta}(\mathbf{X})\mathbf{B} + \varepsilon$ is (for a 2D case):

$$\dot{\mathbf{X}} = \underbrace{\begin{bmatrix} \dot{x}_1(t_1) & \dot{x}_2(t_1) \\ \dots & \dots \\ \dot{x}_1(t_n) & \dot{x}_2(t_n) \end{bmatrix}}_{\substack{\text{matrix of time derivatives} \\ \text{(computed numerically)}}}, \quad \mathbf{B} = \underbrace{\begin{bmatrix} \beta_{01} & \beta_{02} \\ \dots & \dots \\ \beta_{51} & \beta_{52} \end{bmatrix}}_{\substack{\text{sparse matrix of} \\ \text{coeffs. (unknown)}}},$$

$$\mathbf{\Theta}(\mathbf{X}) = \underbrace{\begin{bmatrix} 1 & x_1(t_1) & x_2(t_1) & x_1(t_1)x_2(t_1) & x_1^2(t_1) & x_2^2(t_1) \\ \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & x_1(t_n) & x_2(t_n) & x_1(t_n)x_2(t_n) & x_1^2(t_n) & x_2^2(t_n) \end{bmatrix}}_{\text{polynomials of observed data}}$$

- Common approach to estimation: Estimate each col. of $\mathbf{B}$ via the Lasso: $\hat{\mathbf{B}}_j = \mathrm{argmin}_{\mathbf{B}_j} \|\dot{\mathbf{X}}_j - \mathbf{\Theta}(\mathbf{X})\mathbf{B}_j\|_2^2 + \lambda\|\mathbf{B}_j\|_1$
- Non-zero entries of $\hat{\mathbf{B}}$ indicate which terms belong in $\mathbf{f}$

Recovering $\mathrm{d}\mathbf{x}/\mathrm{d}t = \mathbf{f}(\mathbf{x})$ via Lasso (L_1) regression

Example – Lotka-Volterra simulation: add Gaussian noise (SD = 0.75) to state matrix $\mathbf{X}$ at 48 time points.

L_1 regression (Lasso) recovers the following ODE:

$$\frac{\mathrm{d}x_1}{\mathrm{d}t} = -2.1 + 1.2x_1 - 0.13x_1x_2 + 0.03x_2^2 - 0.0002x_1^3 + 0.0006x_1^2x_2 + \text{more...}$$

$$\frac{\mathrm{d}x_2}{\mathrm{d}t} = 2.3 - 0.08x_1 - 1.7x_2 + 0.04x_1x_2 + 0.0001x_1^3 + 0.001x_2^3$$

The green terms are ones that are actually in the L-V equations: $\frac{\mathrm{d}x_1}{\mathrm{d}t} = \alpha x_1 - \beta x_1 x_2, \quad \frac{\mathrm{d}x_2}{\mathrm{d}t} = \delta x_1 x_2 - \gamma x_2.$

Recovering $\mathrm{d}\mathbf{x}/\,\mathrm{d}t = \mathbf{f}(\mathbf{x})$ via Lasso (L_1) regression

Some issues with using the Lasso to recover dynamics:

- Lasso often does not work well for highly correlated feature matrices. In the SINDy framework, the function library matrix $\mathbf{\Theta}(\mathbf{X})$ has (very) correlated columns.

  Theory for how matrix structure affects Lasso predictive performance:

  [A. Dalalyan, M. Hebiri, J. Lederer; *IEEE Info. Theory* 2012, *Bernoulli* 2017]

- There isn't a widely-accepted notion of statistical significance for Lasso estimates. Here, many terms with small coefficients are included in the learned equations.

Proposed improvements

Recent advances in high-dimensional statistical inference have provided uncertainty quantification for regularized regression.

- Bias-corrected versions of Lasso and ridge regression:
  Hypothesis tests and confidence intervals derived from estimator's asymptotic normality.
  [P. Bühlmann; *Bernoulli*, 2013], [C.-H. Zhang, S. Zhang; *JRSS-B*, 2013], [A. Javanmard, A. Montanari; *JMLR*, 2014]

- SEMMS (Scalable Empirical Bayes Model Selection):
  Bayesian algorithm for sparse variable selection in linear models
  [H. Y. Bar, J. G. Booth, M. T. Wells; *JCGS*, 2020]

Idea: Retain only statistically significant terms provided by these methods in the learned differential equations.

Bias-corrected regularized regression

Usual linear model: $\mathbf{Y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon}$, where $\mathbf{X} \in \mathbb{R}^{n \times p}$.

No explicit formulas for the bias and variance of Lasso estimate of $\boldsymbol{\beta}$.

- Bias-corrected Lasso estimator: [C.-H. Zhang, S. Zhang; 2013]

$$\hat{\mathbf{b}}_j = \hat{\boldsymbol{\beta}}_j + \frac{\mathbf{Z}^{(j)T}(\mathbf{Y} - \mathbf{X}\hat{\boldsymbol{\beta}})}{\mathbf{Z}^{(j)T}\mathbf{X}^{(j)}}, \quad j = 1, ..., p$$

  where $\hat{\boldsymbol{\beta}}$ is the regular Lasso estimator,
  $\mathbf{X}^{(j)}$ is the j^{th} column of $\mathbf{X}$,
  $\mathbf{Z}^{(j)}$ are the residuals from Lasso-regressing $\mathbf{X}^{(j)}$ on $\mathbf{X}^{(-j)}$.

- For $\boldsymbol{\varepsilon} \sim N(0, \sigma^2 \mathbf{I}_n)$, sufficiently sparse $\boldsymbol{\beta}$, and $\lambda \propto \sqrt{\log p/n}$,

$$\frac{1}{\sigma}\sqrt{n}\frac{\mathbf{Z}^{(j)T}\mathbf{X}^{(j)}}{\|\mathbf{Z}^{(j)}\|_2}\left(\hat{\mathbf{b}}_j - \boldsymbol{\beta}_j\right) \xrightarrow{d} N(0,1)$$

  Can use this to get conf. intervals/hypothesis tests for each $\boldsymbol{\beta}_j$.

SEMMS: Bayesian variable selection

Scalable EMpirical Bayes Model Selection

[H. Bar, J. Booth, M. T. Wells, *JCGS* 2020]

- Method places a 3-component Gaussian mixture prior on regression coefficients, indicating that each feature (polynomial term) has a positive, negative, or zero effect on the outcome (time derivative)
- I.e., the method estimates the sign of each feature
- Uses computationally efficient generalized alternating minimization algorithm
- Inference: fit standard linear regression model to the non-zero features and use usual confidence intervals

Simulation: the Van der Pol system

A second-order ODE:

$$\frac{\mathrm{d}^2 x}{\mathrm{d}t} = -x + \mu \frac{\mathrm{d}x}{\mathrm{d}t} - \mu x^2 \frac{\mathrm{d}x}{\mathrm{d}t}.$$

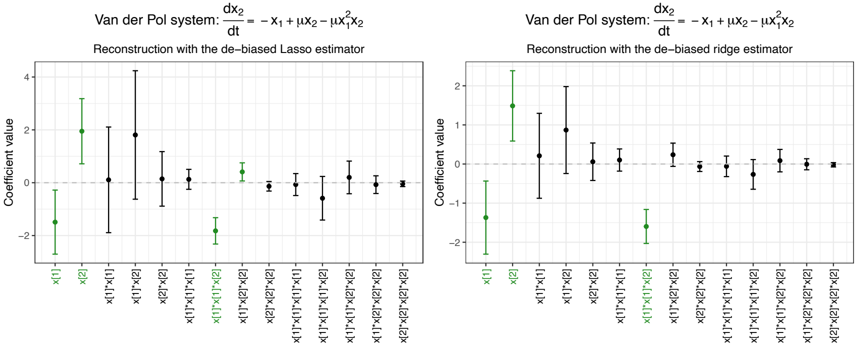
Write as a system of two first-order equations:

$$\begin{aligned}
\frac{\mathrm{d}x_1}{\mathrm{d}t} &= x_2 \\
\frac{\mathrm{d}x_2}{\mathrm{d}t} &= -x_1 + \mu x_2 - \mu x_1^2 x_2
\end{aligned}$$

Simulation: Add noise to the numerical solution, then use sparse regression methods to recover the $\mathrm{d}x_2 / \mathrm{d}t$ equation

Results on the Van der Pol equations

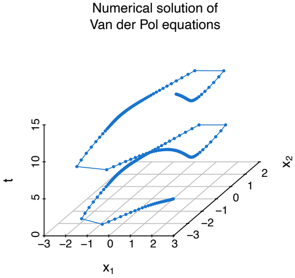
In simulations, including only the statistically significant terms yields sparser ODEs that are much closer to the true equations:

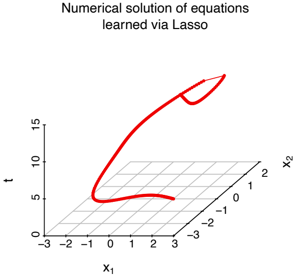


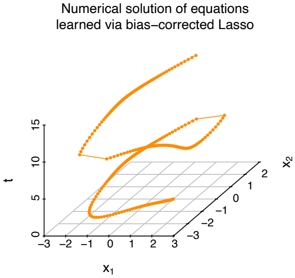
Bias-corrected Lasso result: $\frac{dx_2}{dt} = 0.03 - 1.5x_1 + 1.9x_2 - 1.8x_1^2x_2 + 0.4x_1x_2^2$

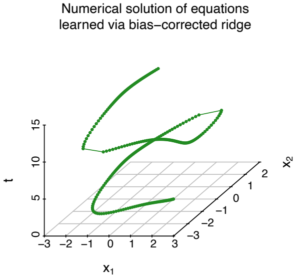
Bias-corrected ridge result: $\frac{dx_2}{dt} = 0.03 - 1.4x_1 + 1.5x_2 - 1.6x_1^2x_2$

Results on the Van der Pol equations









Thank you!

Sara Venkatraman
skv24@cornell.edu

https://sara-venkatraman.github.io