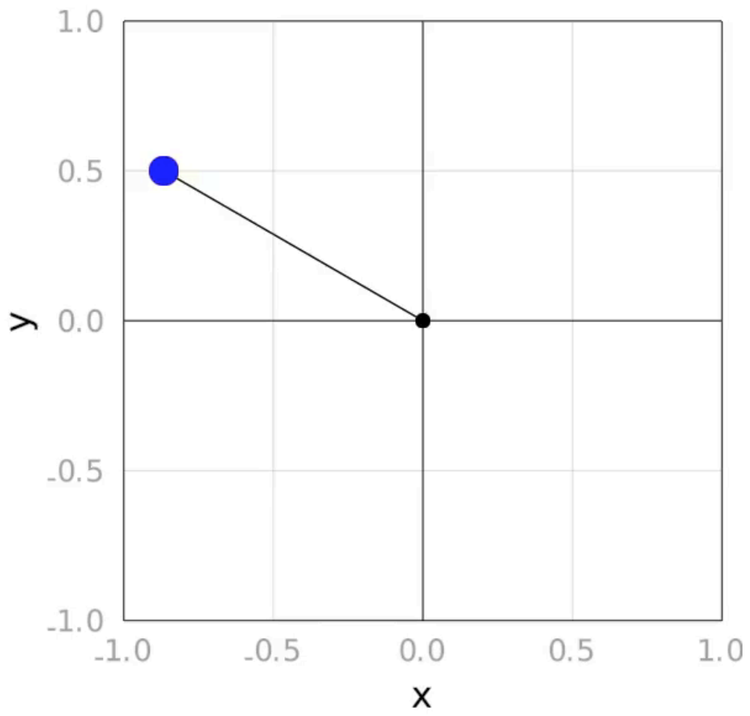
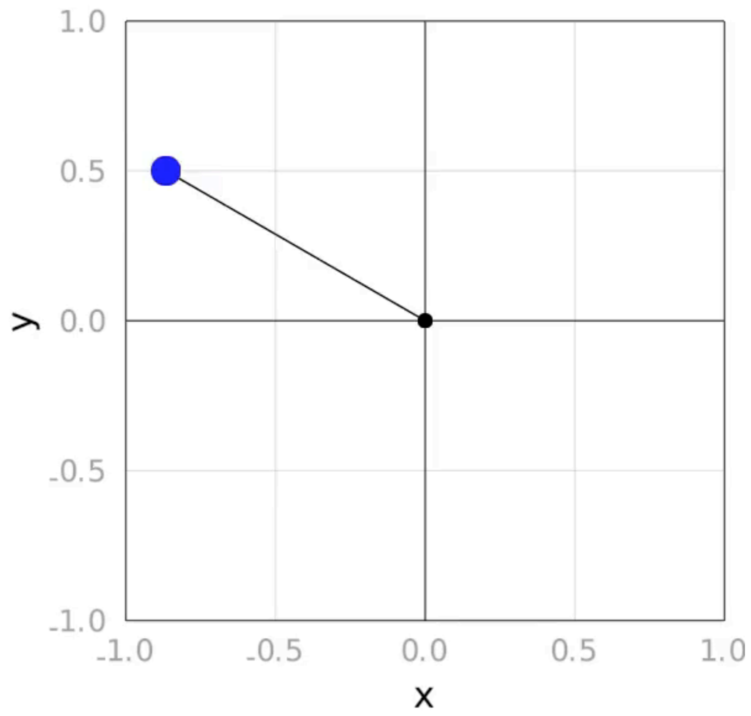
Using Data to Discover Dynamics



Alasdair Hastewell

MIT Applied Math

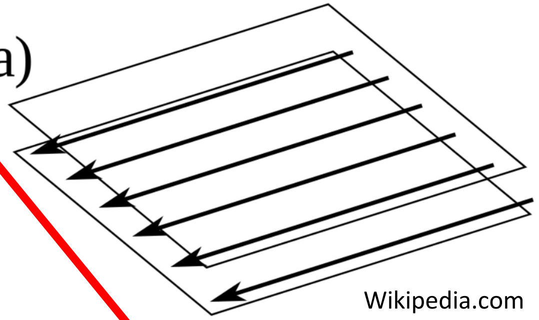


The traditional way

Simplify with assumptions

$$\frac{d^2 u}{dy^2} = -1; \quad u(0) = u(1) = 0$$

a)



Wikipedia.com

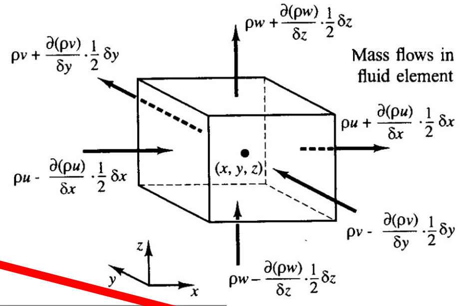
Observe – experiment

by Frits Ahlefeldt

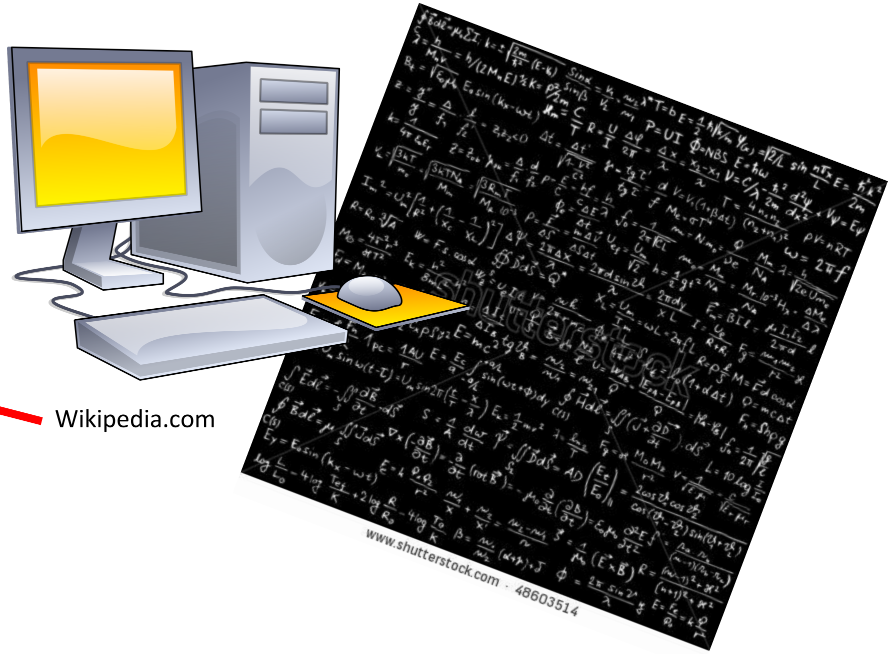
Start with first principles/conservation laws

$$\frac{d}{dt}\int_{\Omega} \phi \, d\Omega = -\int_{\Gamma} \phi \mathbf{u} \cdot \mathbf{n} \, d\Gamma - \int_{\Omega} s \, d\Omega$$

$$F = ma$$

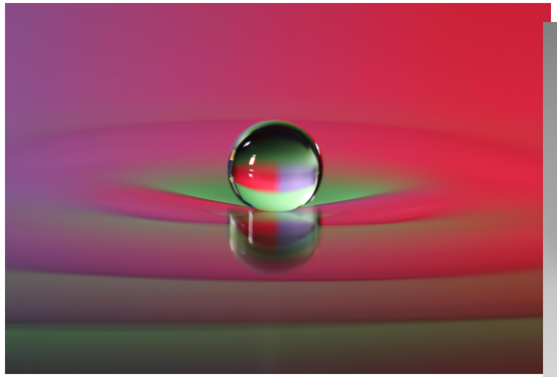


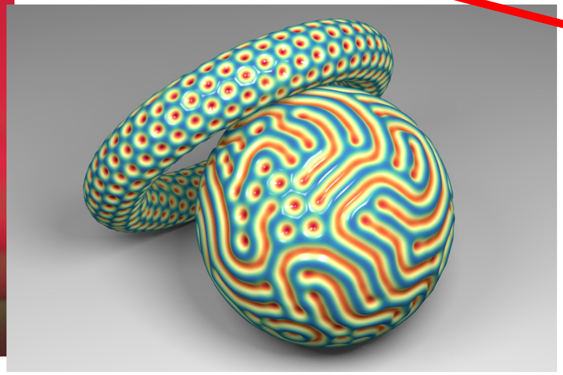
Solve equations



Wikipedia.com

Compare to experiment

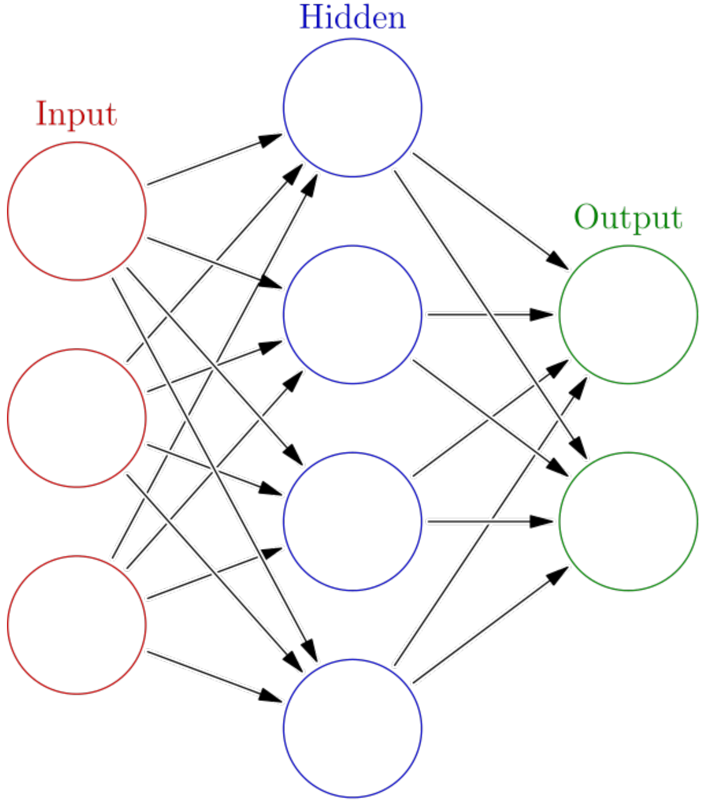




Big data modern approach

Data

Train model



Wikipedia.com

Predict and compare

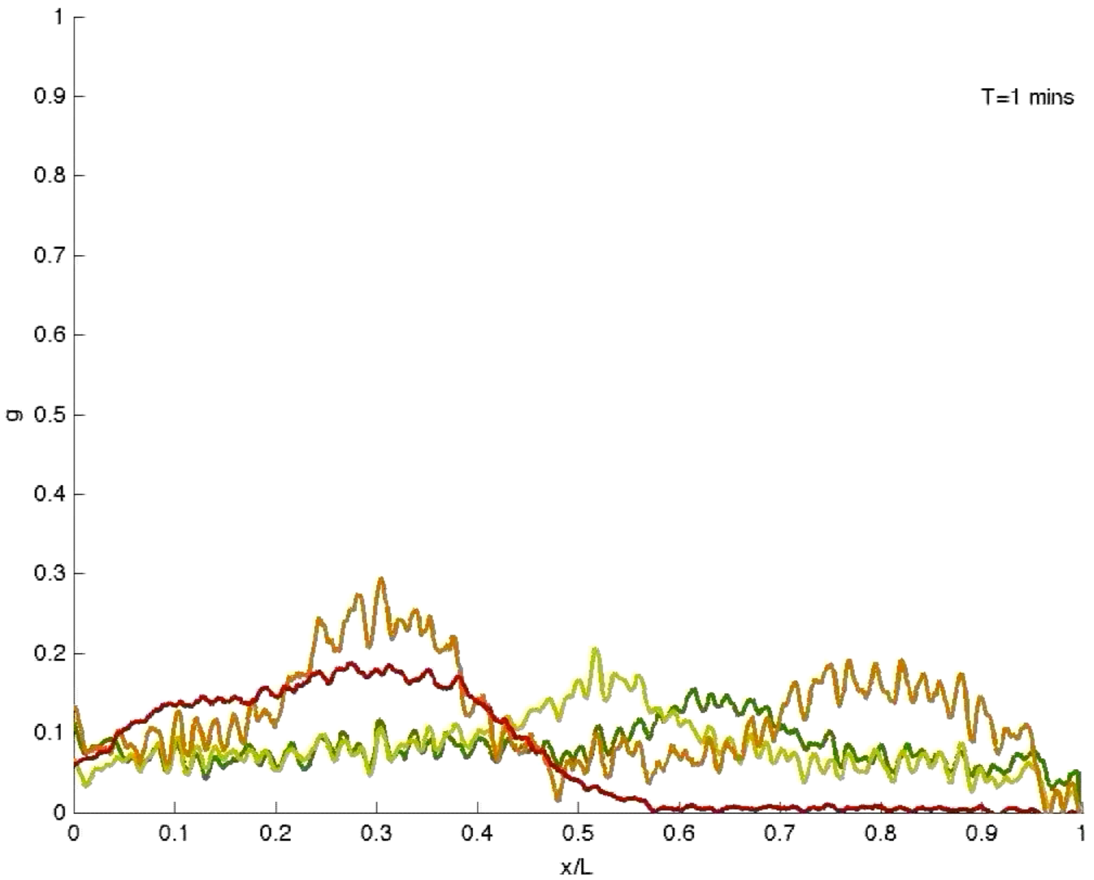
The traditional way is slow

- Many complicated systems are hard to derive from first principles
- The rate of data generation faster than analysis

Equations provide insights

- Machine learning provides weights
  - Hard for a human to interpret
  - Relation to existing models?

Dynamics of the four main gap gene expression profiles – *Drosophila*

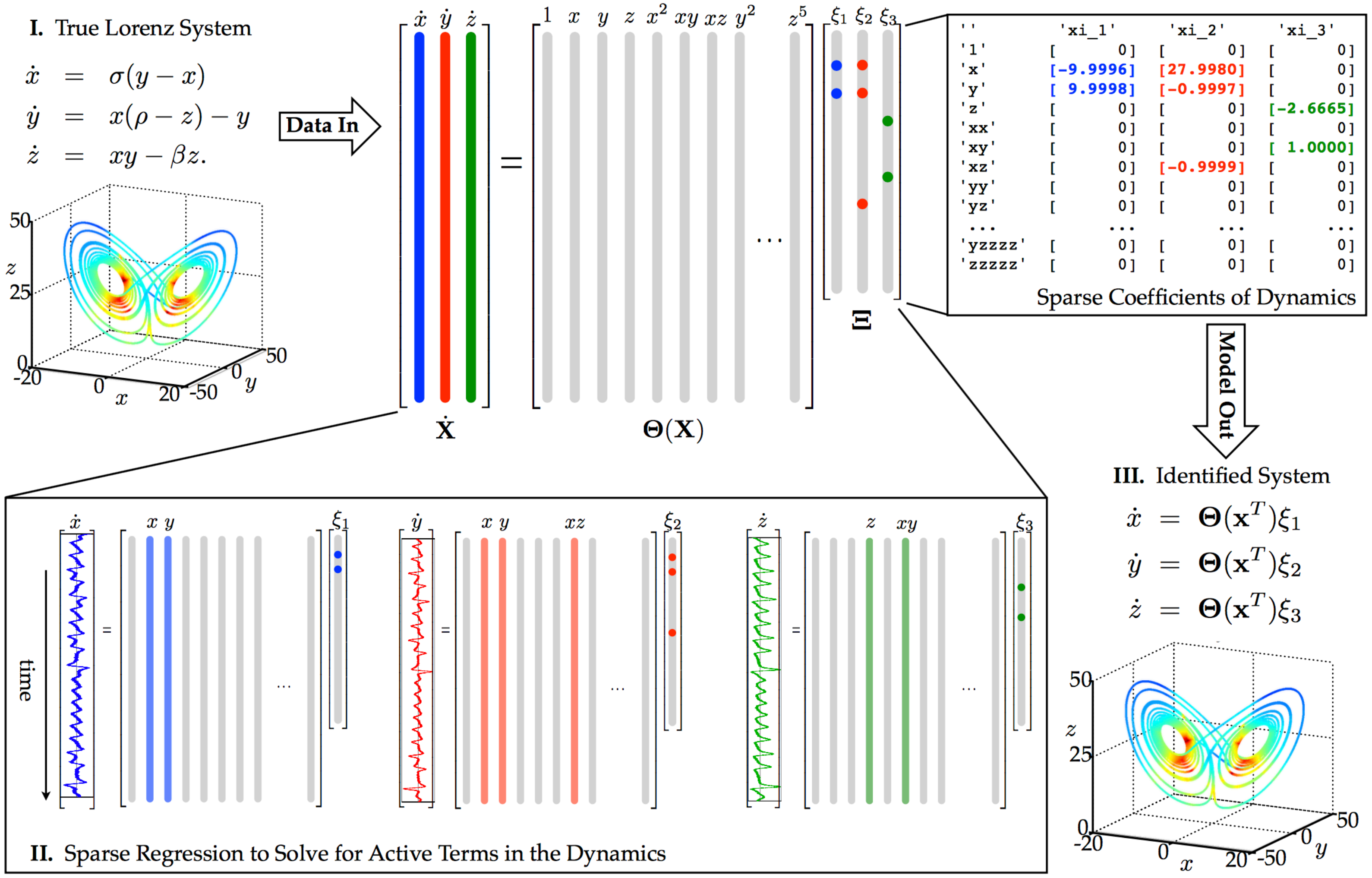


Dubuis JO, Samanta R, Gregor T (2013) Accurate measurements of dynamics and reproducibility in small genetic networks. Molecular Systems Biology 9: 639.

Where do these ideas intersect

- Use data to “derive” equations
- Clearer links to established theory
- Increased speed of analysis and prediction

The big idea – library of possible terms



- Produce a library of possible terms
- Use regression to find estimates for the coefficients
- Reintegrate to compare

S. Brunton, J. Proctor and J. N. Kutz, Discovering governing equations from data: sparse identification of nonlinear dynamical systems arXiv:1509.03580

Sparsity triumphs

- In most physical models only have a few coefficients

- Add a penalty term for number of coefficients

$$\hat{\xi} = arg\, min_{\xi} ||\mathbf{\Theta}\xi - \mathbf{U}_t||_2^2 + \lambda||\xi||_2^2$$

- Different penalty terms – varying stability

Challenges

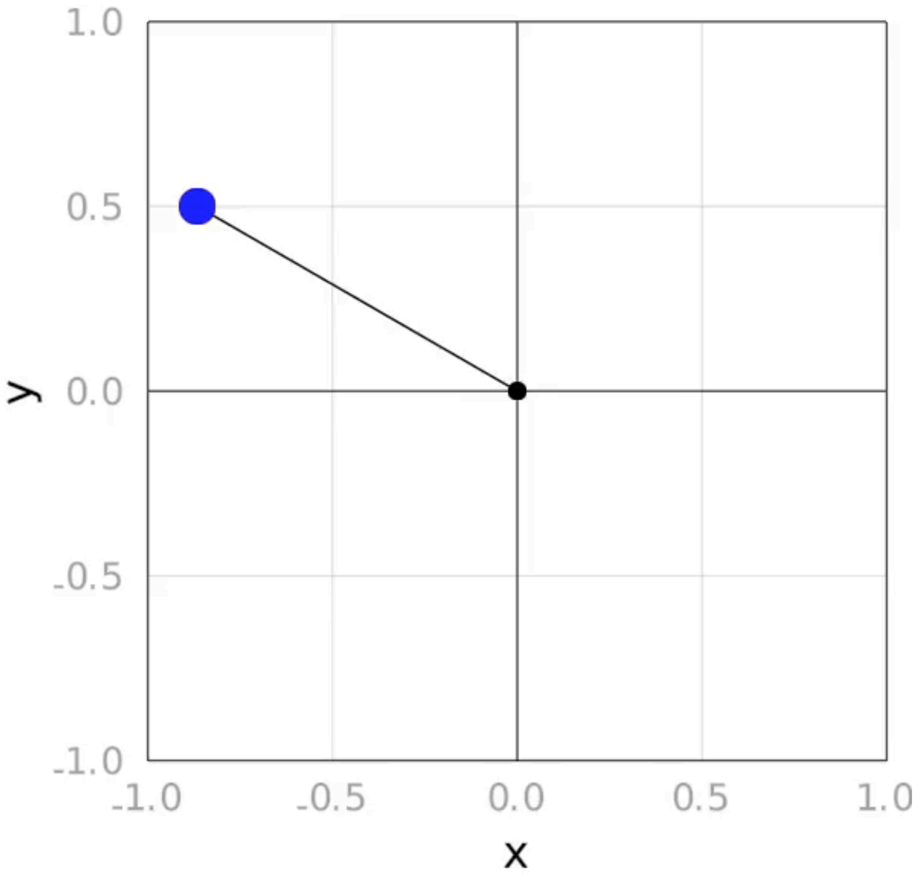
- Data has noise
  - Derivatives amplify noise
  - Algorithm is sensitive
- Often a limited amount of data
- Some codes exist but are scattered
  - MATLAB
  - Python
  - Convoluted to use

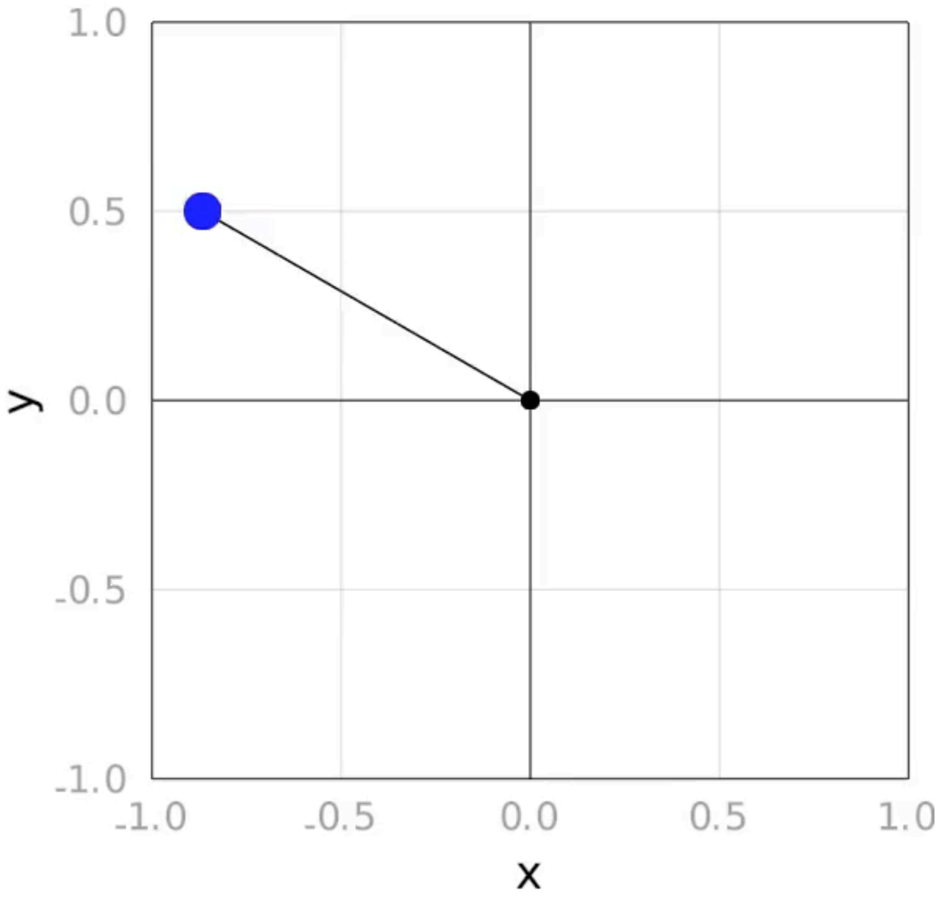
Noisy derivatives

- ApproxFun
  - Evenly spaced data – not supported
  - Expansions are smooth
  - Derivatives efficient to compute
  - Coefficient thresholding can smooth

- Wavelets
  - Over determined more robustness
  - Smooth function -> Finite Difference

With noise – carefully selected parameters





Timings compared to MATLAB

Function	MATLAB	Julia
Making Library	6.309 seconds	0.495292 seconds
Regressing Dynamics	0.362 seconds	0.771896 seconds
Reading Coefficients	0.012 seconds	0.000128 seconds
Total	6.683 seconds	1.267 seconds

~ 5x speed up in Julia

Conclusions

- Computers can be used to help in discovery of dynamical systems
  - More robustness is needed to noise
- Julia provides significant speed up over MATLAB for these problems
  - Broadcasting allows for concise code
  - Multiple dispatch and typing

References

- Packages used
  - ApproxFun, DifferentialEquations

- S. Brunton, J. Proctor and J. N. Kutz, Discovering governing equations from data: sparse identification of nonlinear dynamical systems arXiv:1509.03580

- *Data-driven identification of parametric partial differential equations*, arXiv:1806.00732 (Rudy, Brunton & Kutz)

- M. S. Floater and K. Hormann, Barycentric rational interpolation with no poles and high rates of approximation, *Numerische Mathematik* 107 (2007), 315—331

- Accurate measurements of dynamics and reproducibility in small genetic networks, Julien O. Dubuis, Reba Samanta and Thomas Gregor, *Molecular Systems Biology* **9**: 639 (2013)