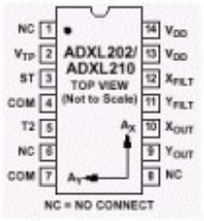
ENM 540: Data-driven modeling and probabilistic scientific computing

Lecture #1: Introduction to data-driven modeling

Paris Perdikaris

January 11th, 2018

Roadmap to Trillion Sensors



Accelerometer


Gyro


Pendulum Resistive
Tilt Sensors


Piezo Bend Sensor

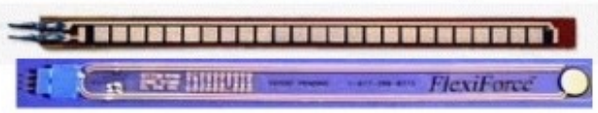

Metal Detector

Gas Sensor


Gieger-Muller
Radiation Sensor


Digital Infrared Ranging

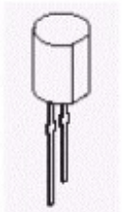

CDS Cell
Resistive Light Sensor

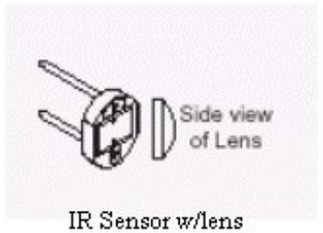


Resistive Bend Sensors


UV Detector


Pyroelectric Detector


IR Pin
Diode

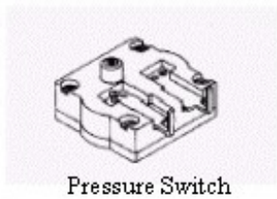


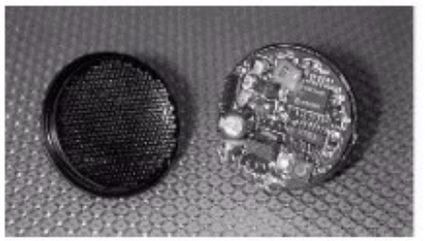
IR Sensor w/lens


Limit Switch


Mechanical Tilt Sensors

Touch Switch


Pressure Switch


Miniature Polaroid Sensor


IR Reflection
Sensor

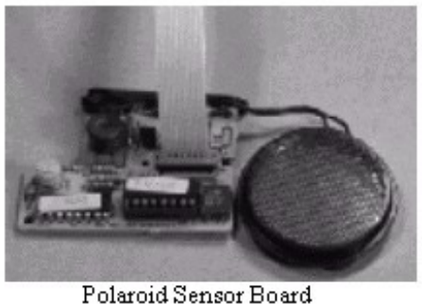

IR Amplifier Sensor


Thyristor

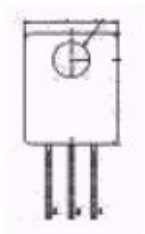

Magnetic Sensor


Magnetic Reed Switch


Hall Effect
Magnetic Field
Sensors


Polaroid Sensor Board


IRDA Transceiver

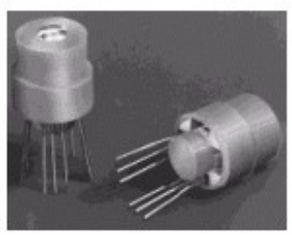

Lite-On IR
Remote Receiver

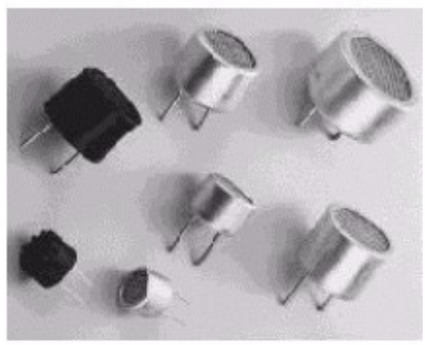

Radio Shack
Remote Receiver


IR Modulator
Receiver


Solar Cell

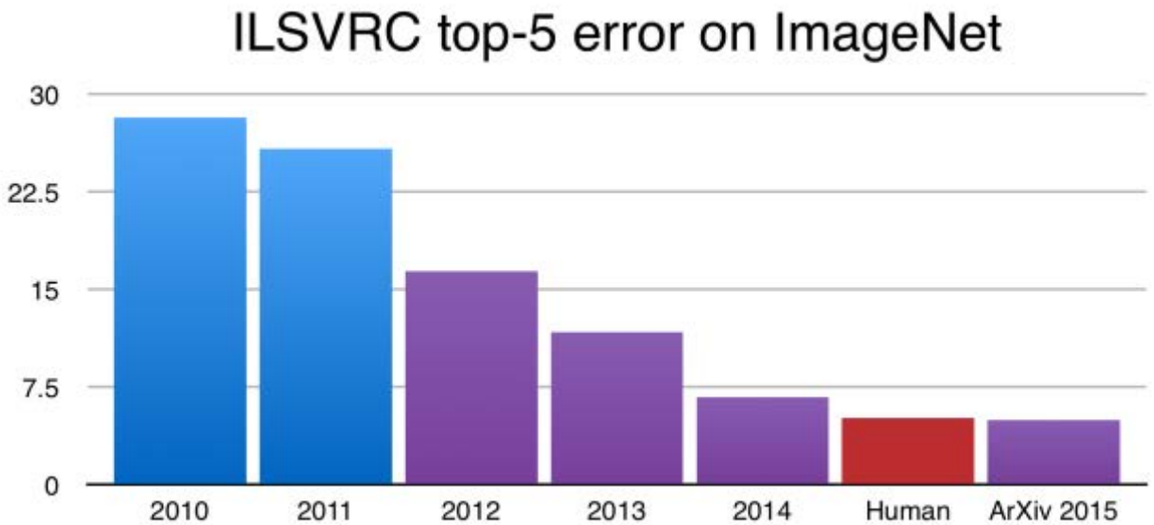

Compass

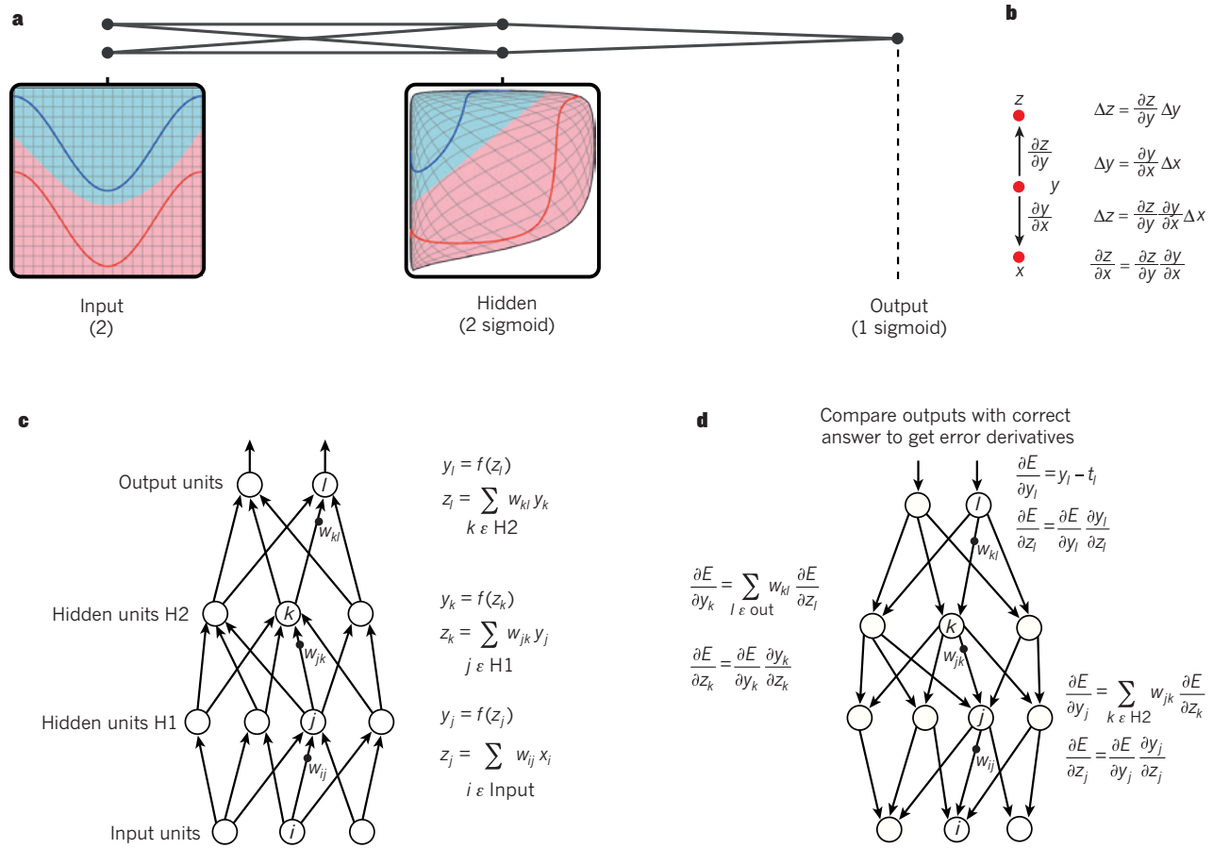

Compass


Piezo Ultrasonic Transducers

Sensors are expected to generate a bronto-bytes (1000 trillion trillion) of data by 2025!

Recent success of machine learning





All models are wrong

but some are useful

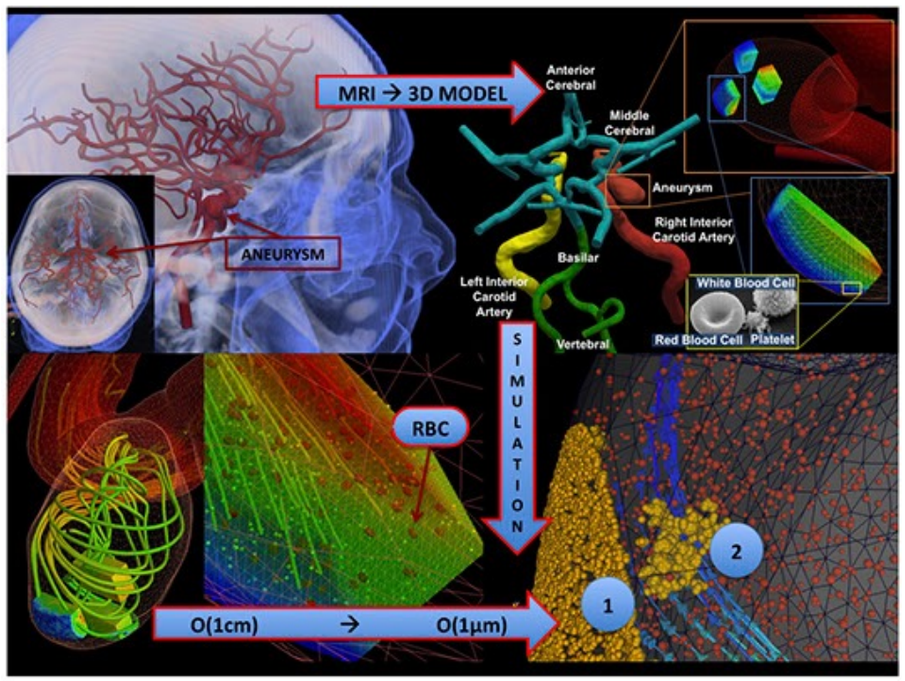
George E.P. Box

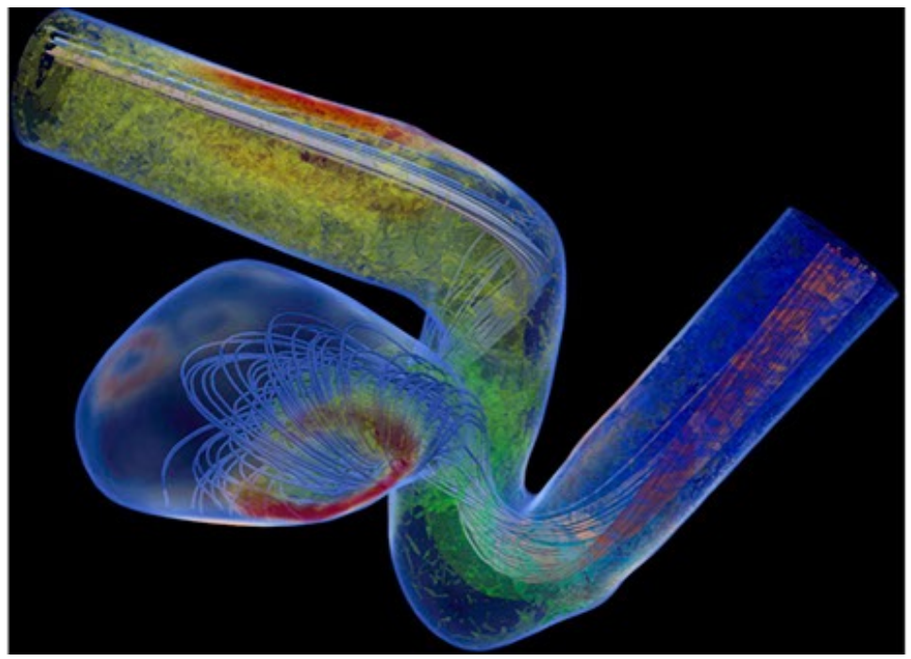
*image credit: S. Sherwin (Imperial)

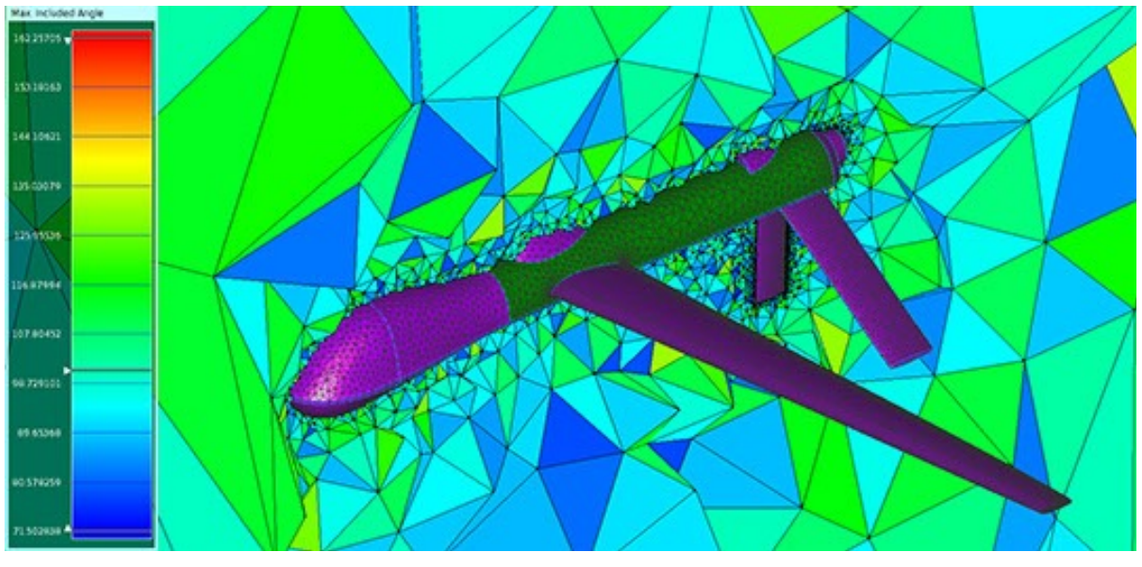
$$\frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \nabla)\boldsymbol{u} = -\frac{1}{\rho}\nabla p + \nu \nabla^2 \boldsymbol{u}$$

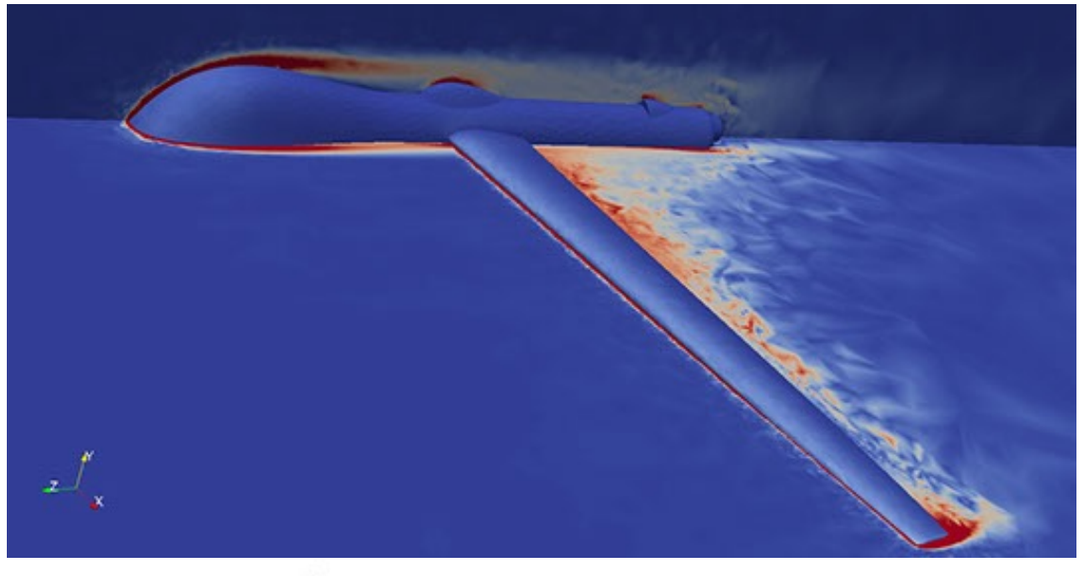
$$\nabla \cdot \boldsymbol{u} = 0$$

Computational science & engineering



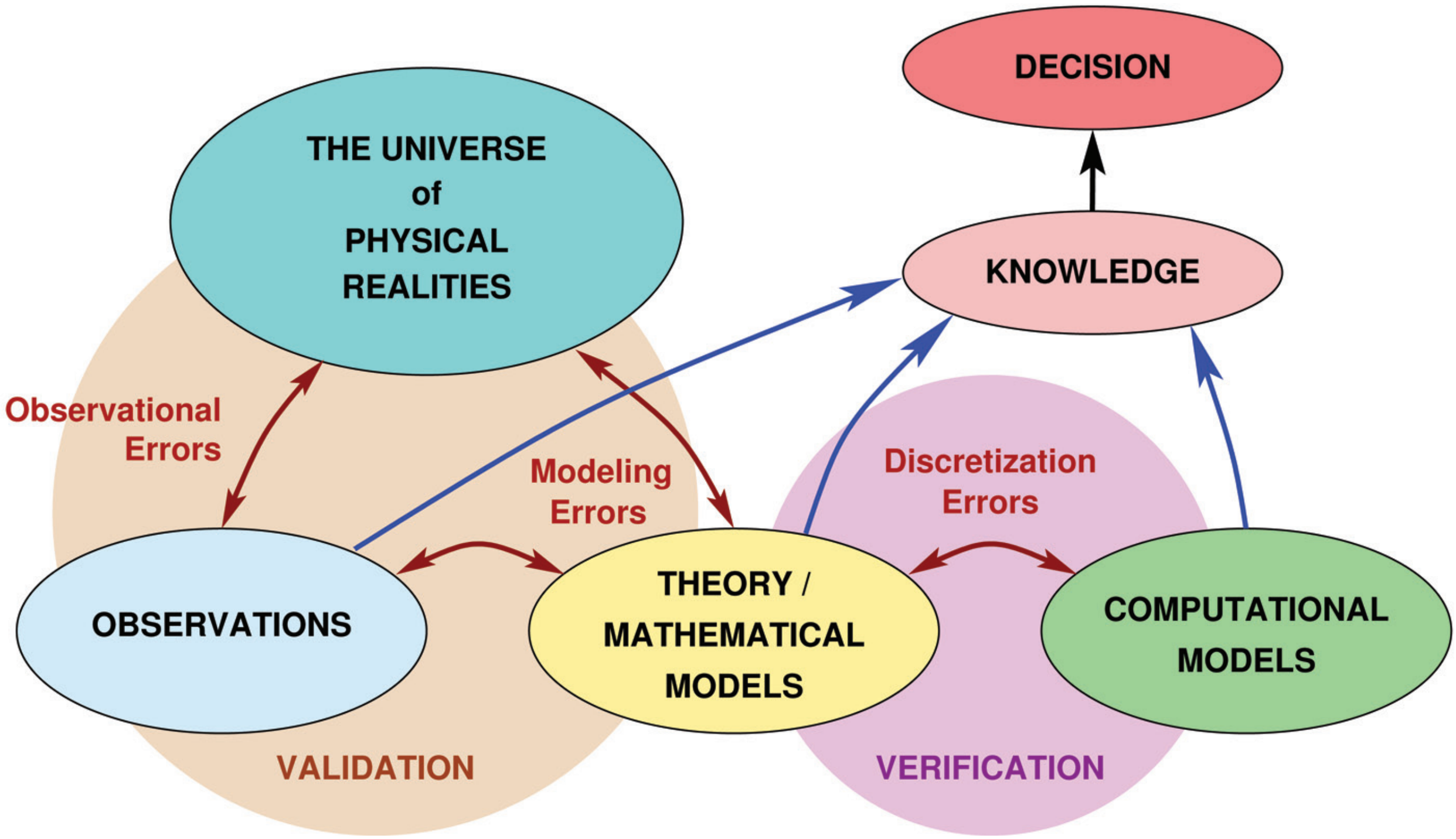






One exaflop is a thousand petaflops or a quintillion (10^18) floating point operations per second!

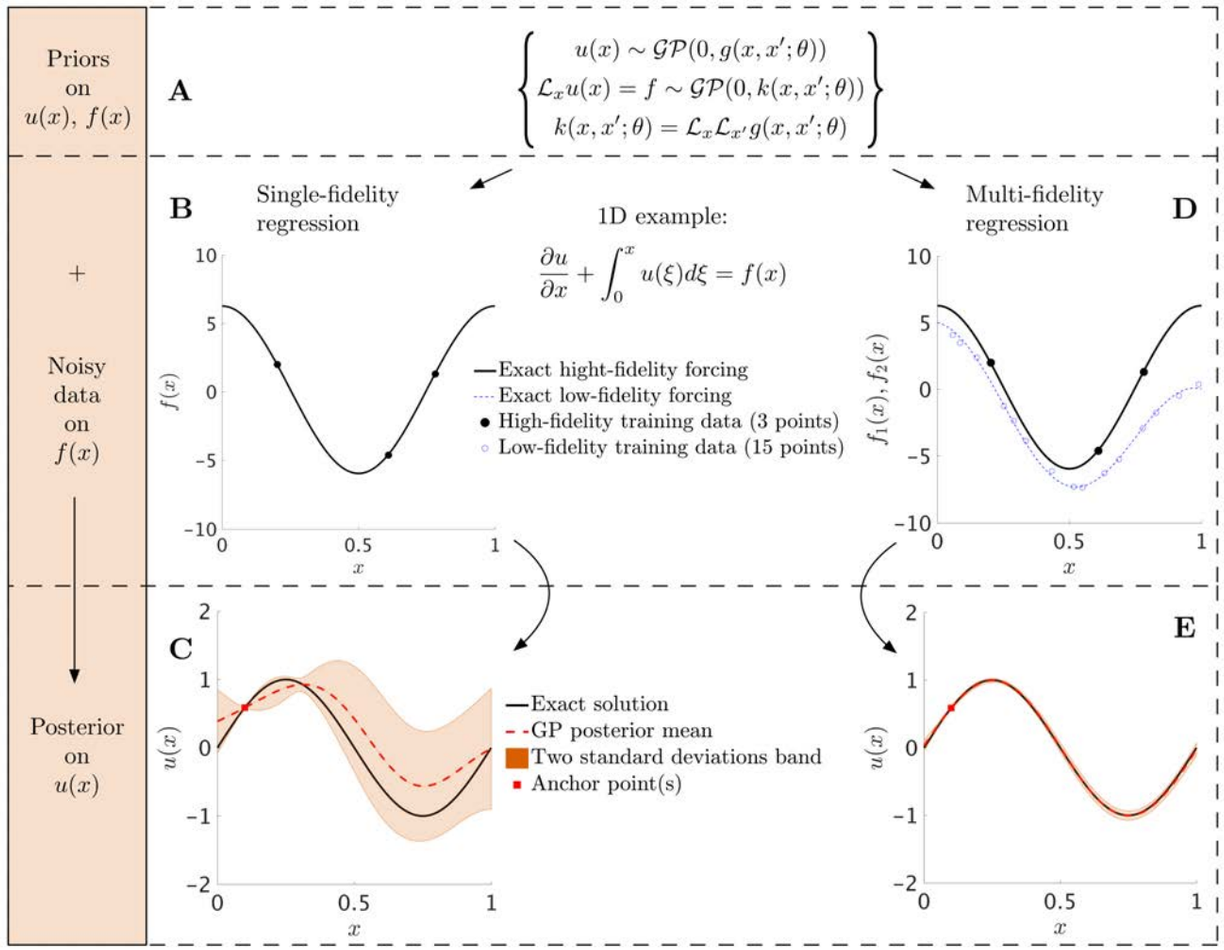
Predictive science



Oden, T., Moser, R., & Ghattas, O. (2010). Computer predictions with quantified uncertainty, part I. SIAM News, 43(9), 1-3.

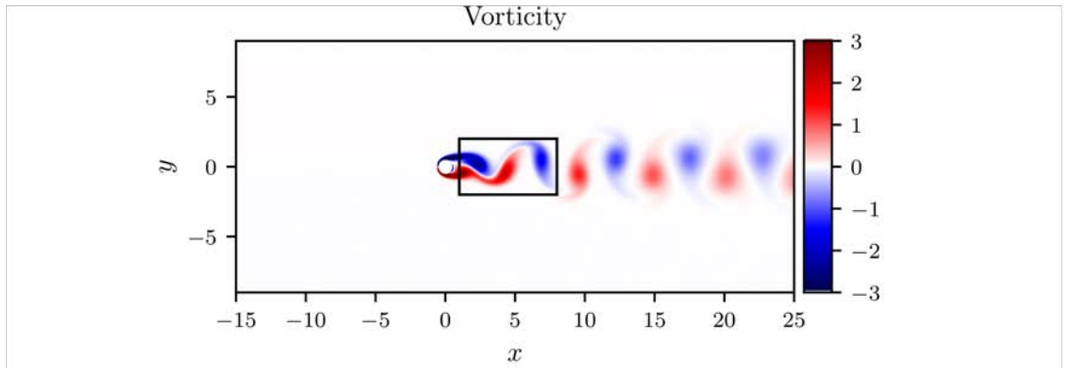
CSE for ML

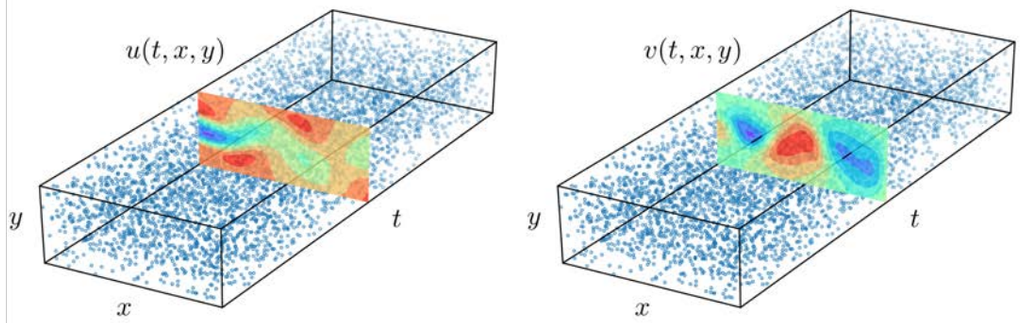
Leverage domain-specific prior information to construct more structured, interpretable, and accurate and data-efficient learning algorithms.

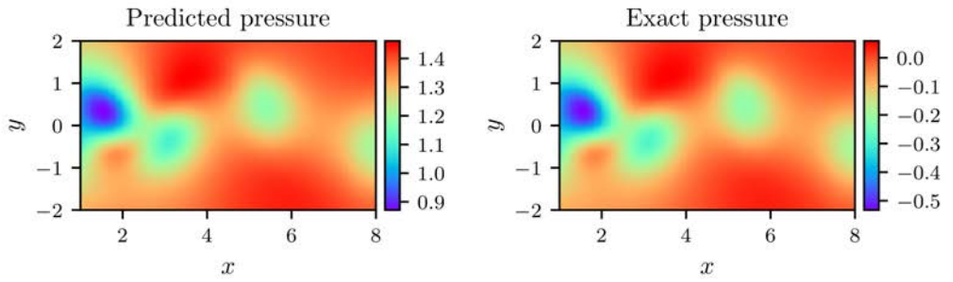


ML for CSE

Leverage recent developments in machine learning to construct robust data-driven solvers, discover new models, and address longstanding challenges in scientific computing (e.g. high-dimensional problems, UQ, etc.)







Correct PDE	$u_t + (uu_x + vu_y) = -p_x + 0.01(u_{xx} + u_{yy})$ $v_t + (uv_x + vv_y) = -p_y + 0.01(v_{xx} + v_{yy})$
Identified PDE (clean data)	$u_t + 0.999(uu_x + vu_y) = -p_x + 0.01047(u_{xx} + u_{yy})$ $v_t + 0.999(uv_x + vv_y) = -p_y + 0.01047(v_{xx} + v_{yy})$
Identified PDE (1% noise)	$u_t + 0.998(uu_x + vu_y) = -p_x + 0.01057(u_{xx} + u_{yy})$ $v_t + 0.998(uv_x + vv_y) = -p_y + 0.01057(v_{xx} + v_{yy})$

$$f : \mathcal{X} \to \mathcal{Y}$$

Supervised learning

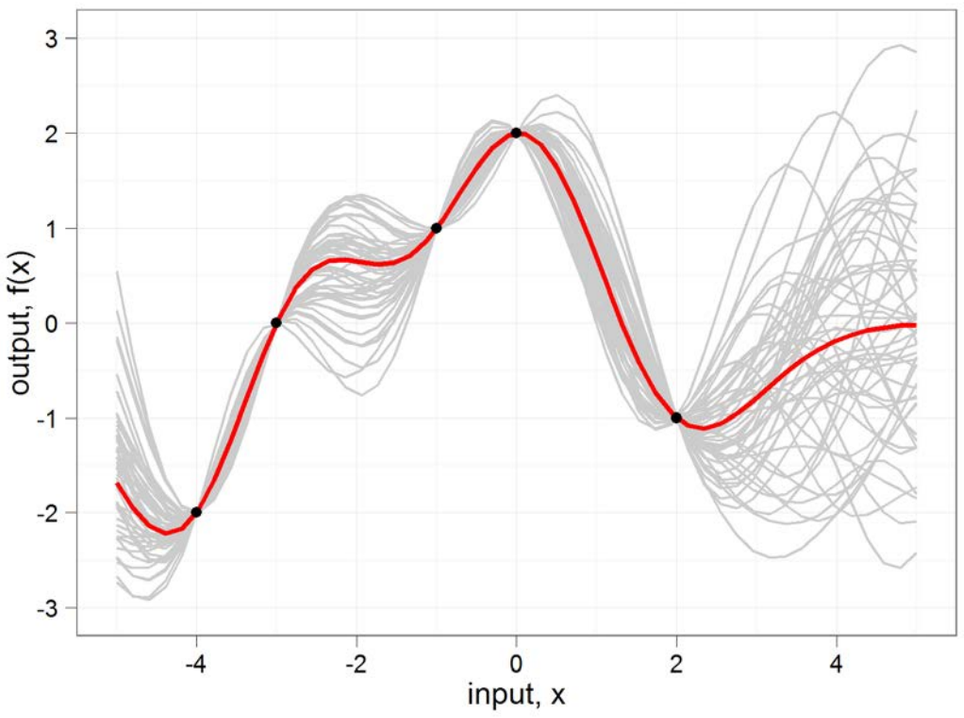
$$f : \mathcal{X} \to \mathcal{Y}$$

$$\mathcal{D} = \{\boldsymbol{x}, \boldsymbol{y}\},\ \boldsymbol{x} \in \mathcal{X},\ \boldsymbol{y} \in \mathcal{Y}$$

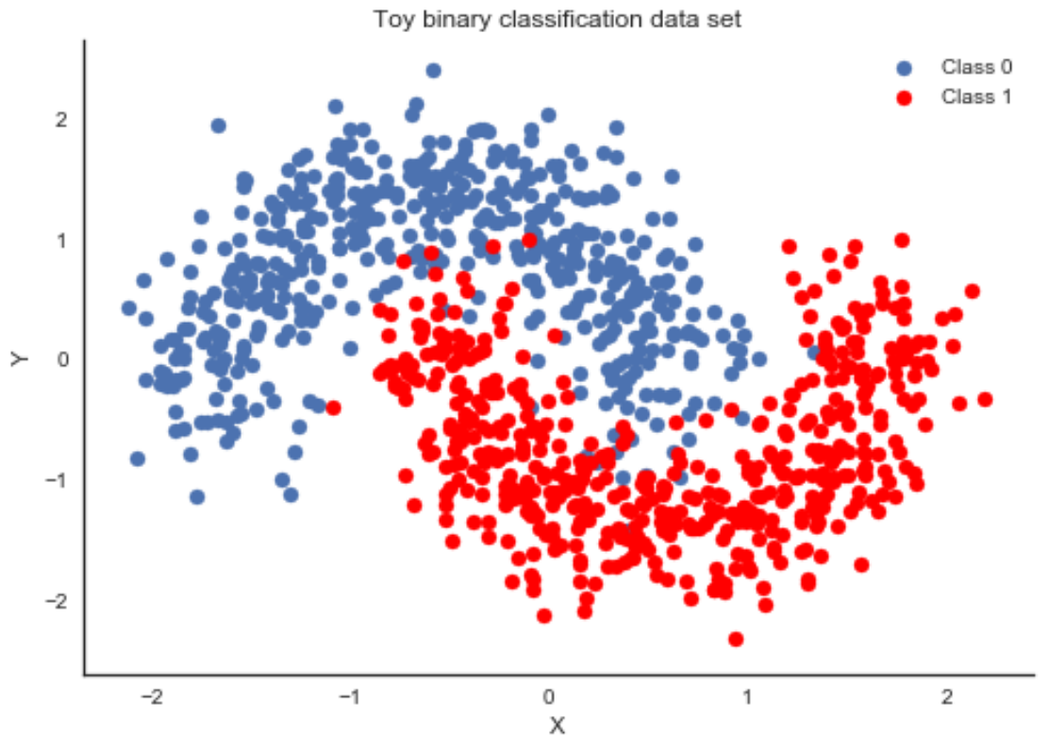
$$\boldsymbol{y} = f(\boldsymbol{x}) + \epsilon$$

$$p(f(\boldsymbol{x}^*)|\boldsymbol{x}^*, \mathcal{D})$$

Regression



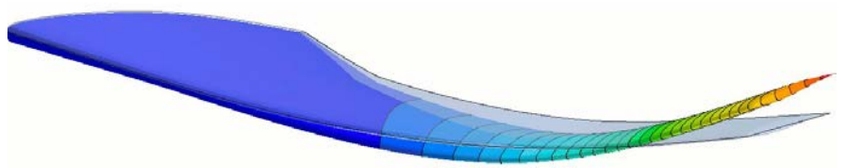
Classification



Regression

A motivating example:

Estimation of wing deflection subject to fluid flow



$$\frac{\partial \sigma_{rr}}{\partial r} + \frac{1}{r}\frac{\partial \sigma_{r\theta}}{\partial \theta} + \frac{\partial \sigma_{rz}}{\partial z} + \frac{1}{r}(\sigma_{rr} - \sigma_{\theta\theta}) + F_r = \rho \frac{\partial^2 u_r}{\partial t^2}$$

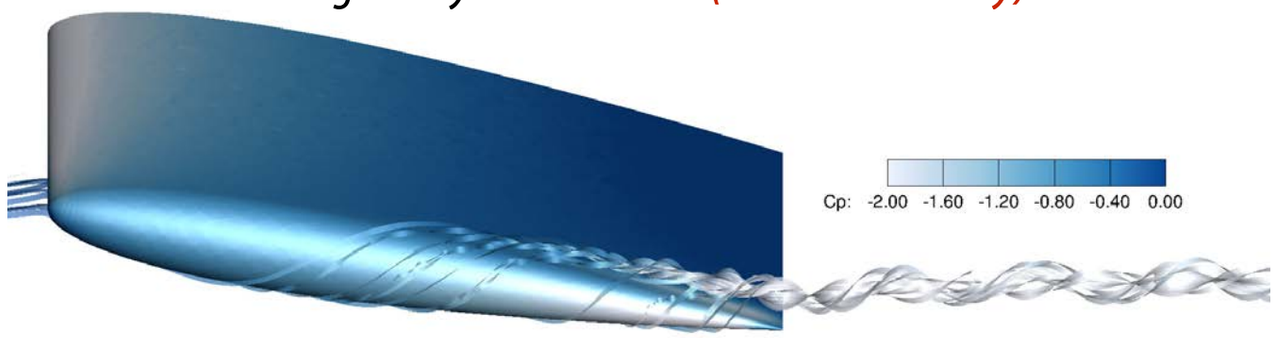
$$\frac{\partial \sigma_{r\theta}}{\partial r} + \frac{1}{r}\frac{\partial \sigma_{\theta\theta}}{\partial \theta} + \frac{\partial \sigma_{\theta z}}{\partial z} + \frac{2}{r}\sigma_{r\theta} + F_\theta = \rho \frac{\partial^2 u_\theta}{\partial t^2}$$

$$\frac{\partial \sigma_{rz}}{\partial r} + \frac{1}{r}\frac{\partial \sigma_{\theta z}}{\partial \theta} + \frac{\partial \sigma_{zz}}{\partial z} + \frac{1}{r}\sigma_{rz} + F_z = \rho \frac{\partial^2 u_z}{\partial t^2}$$

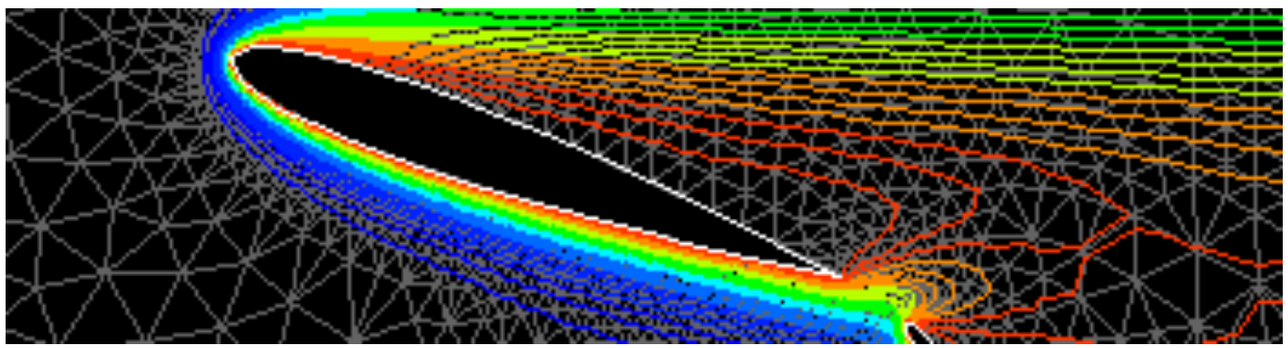
Multi-fidelity data

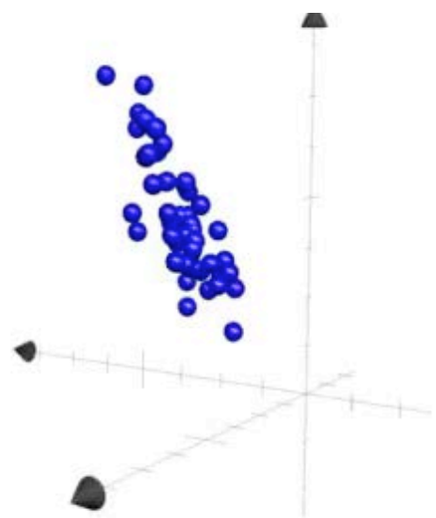
Sensor measurements (high-fidelity)

Large eddy simulations (medium-fidelity)

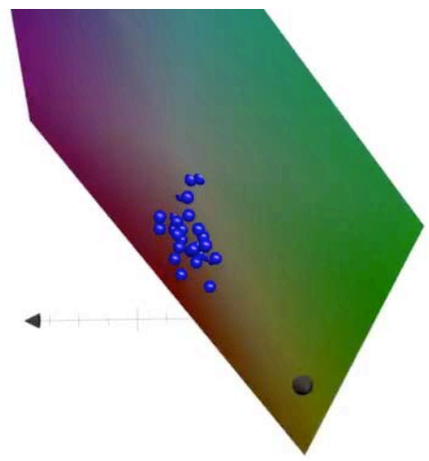


Potential flow simulations (low-fidelity)



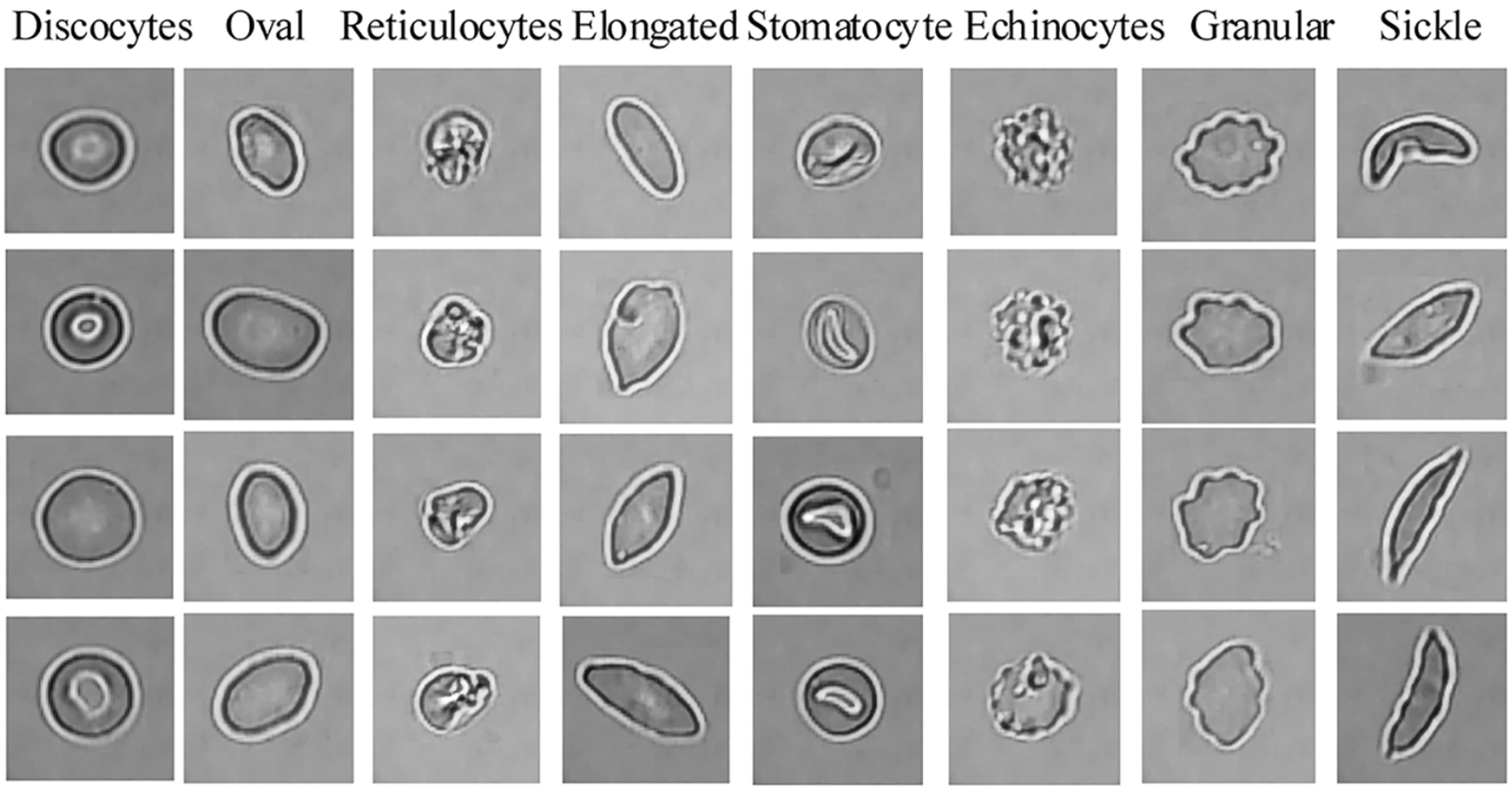


$$\mathcal{D} = \{\boldsymbol{x}, \boldsymbol{y}\},\ \boldsymbol{x} \in \mathcal{X},\ \boldsymbol{y} \in \mathcal{Y}$$

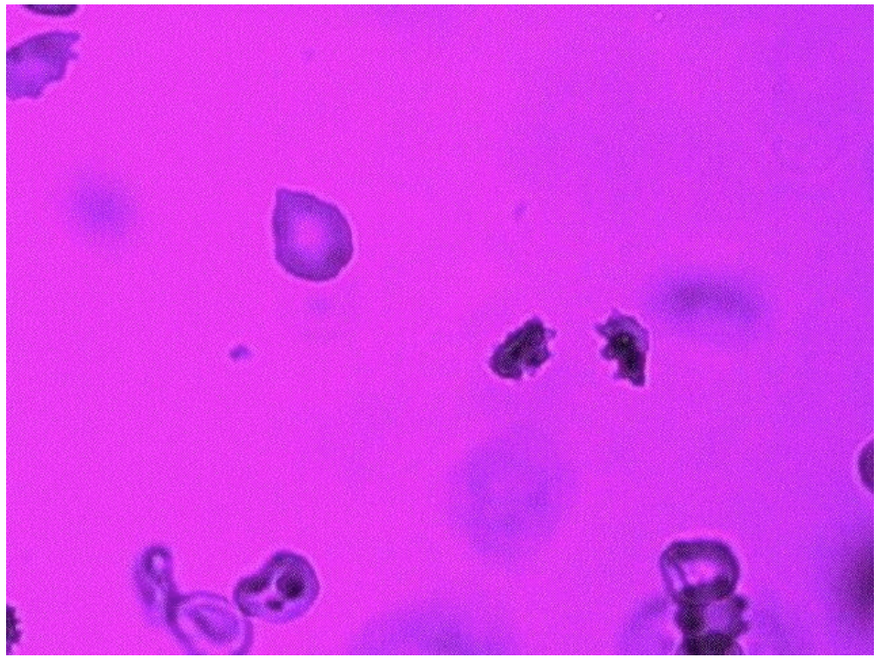


$$\boldsymbol{y} = f(\boldsymbol{x}) + \epsilon$$

Classification



$$\mathcal{D} = \{\boldsymbol{x}, \boldsymbol{y}\},\ \boldsymbol{x} \in \mathcal{X},\ \boldsymbol{y} \in \mathcal{Y}$$



$$\boldsymbol{y} = f(\boldsymbol{x}) + \epsilon$$

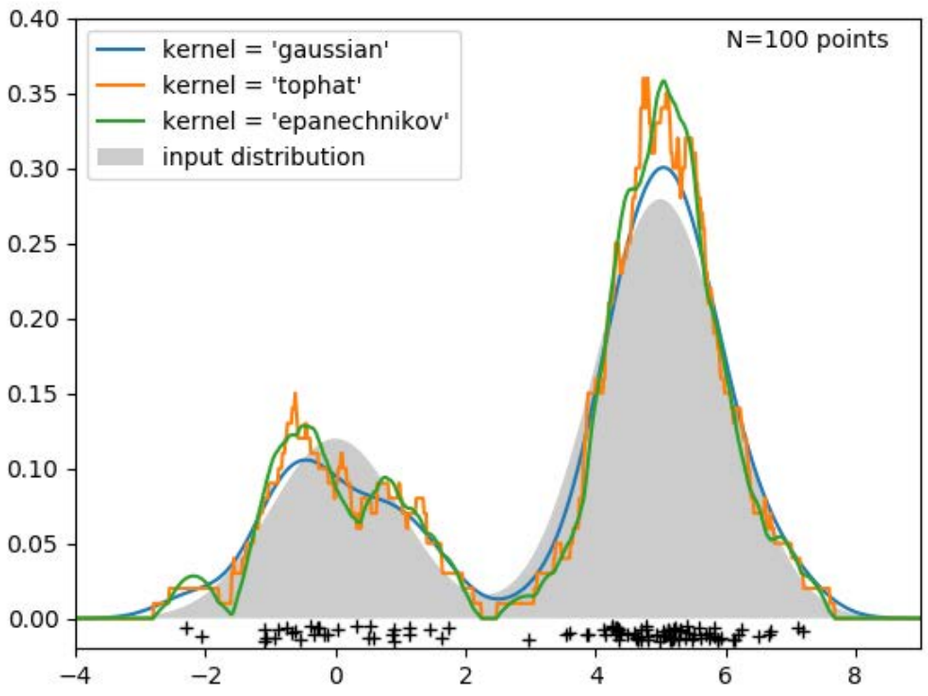
Unsupervised learning

$$\{\boldsymbol{y}\},\ \boldsymbol{y} \in \mathcal{Y}$$

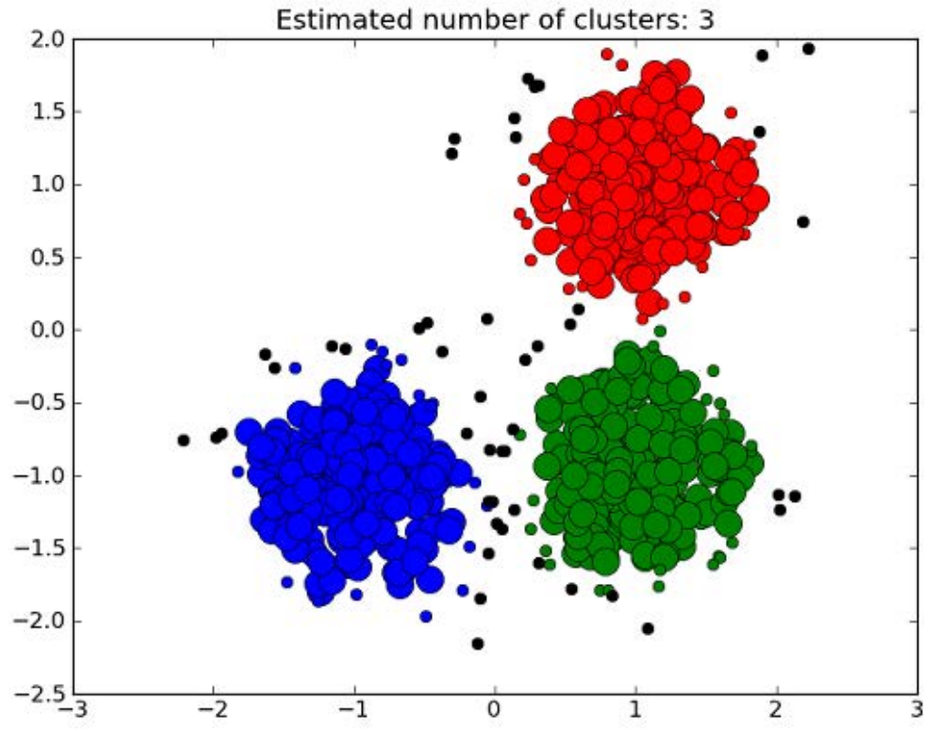
$$\boldsymbol{y} = f(\boldsymbol{z}) + \epsilon$$

$$p(\boldsymbol{y}),\ \boldsymbol{z} \sim p(\boldsymbol{z})$$

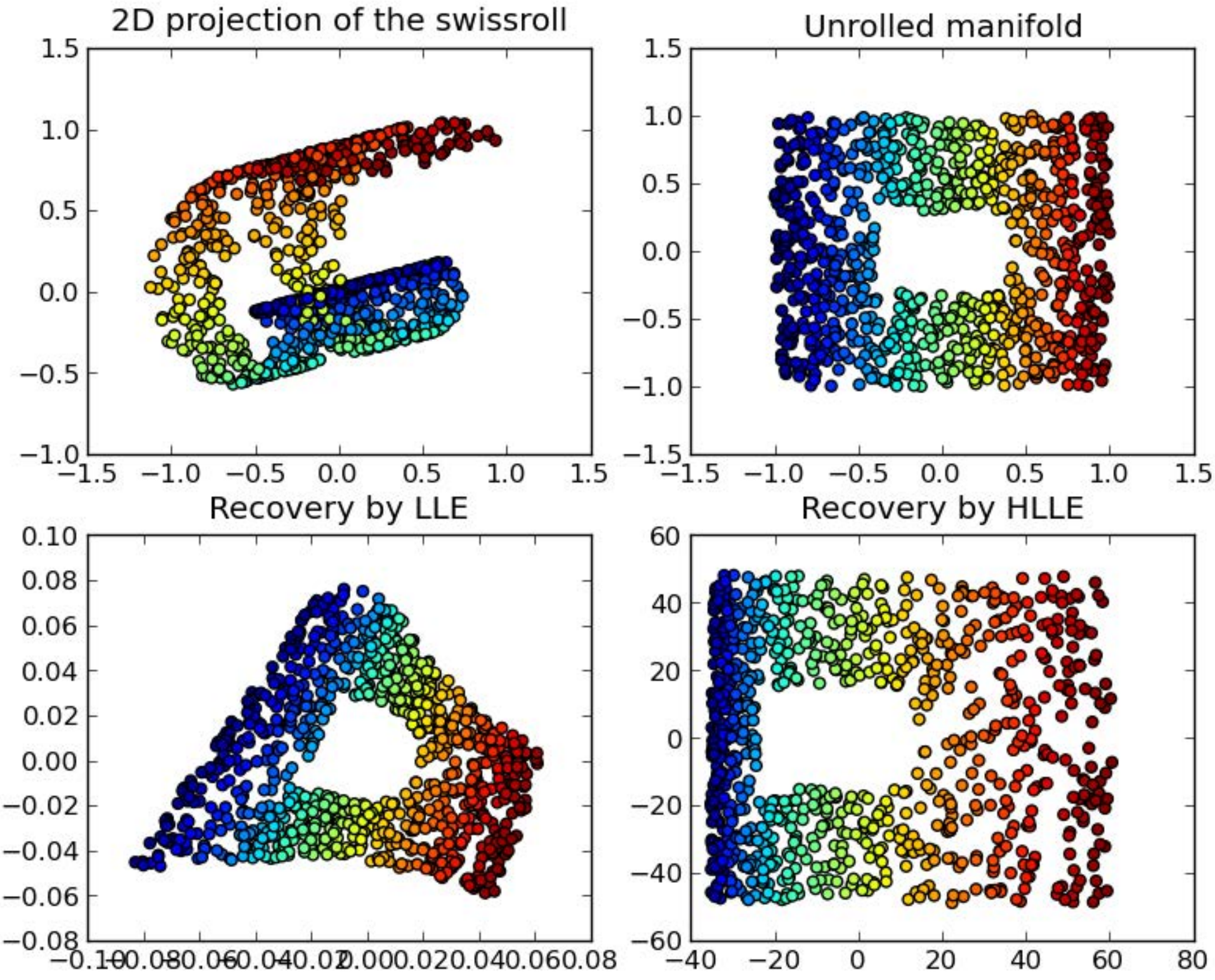
Density estimation



Clustering



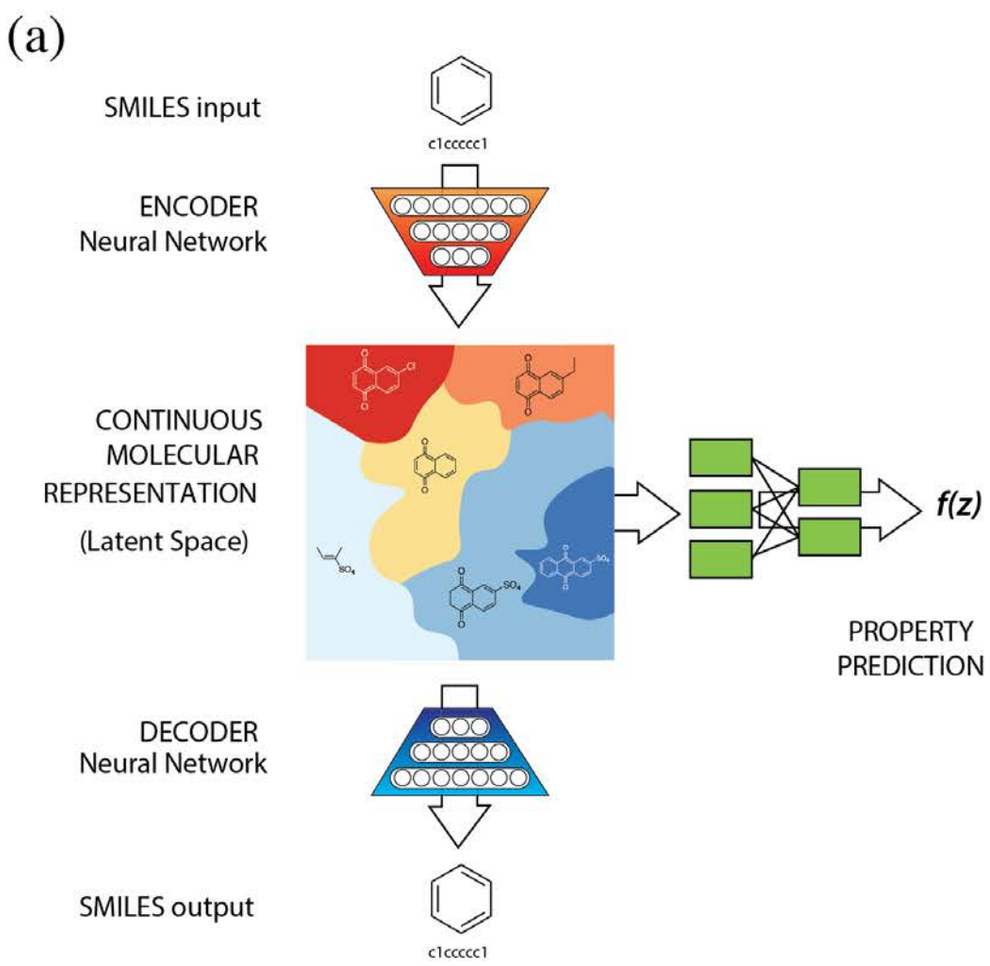
Dimensionality reduction

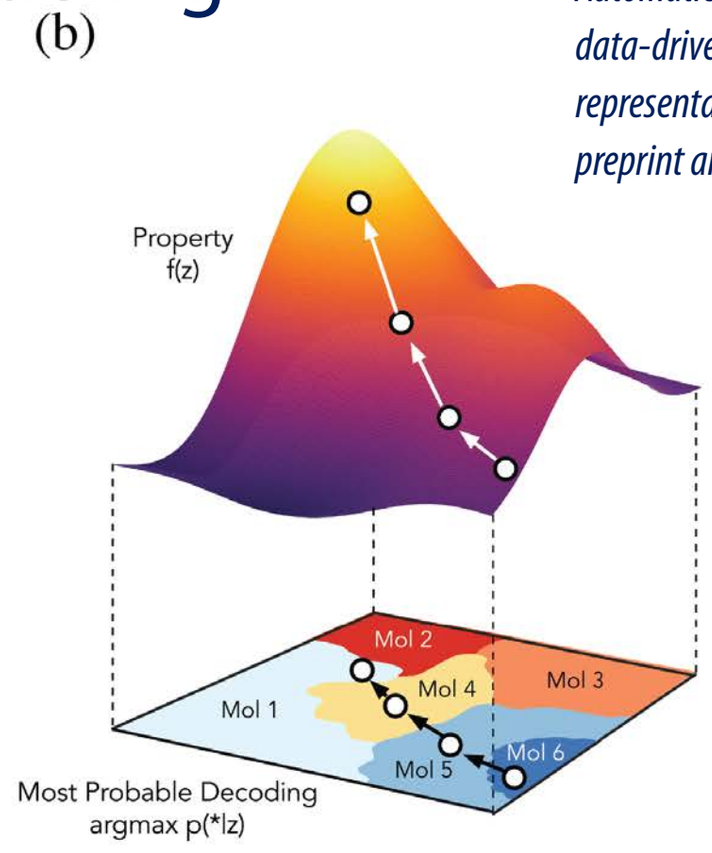


Generative modeling

Generative modeling

Gómez-Bombarelli, R., el. al. (2016). Automatic chemical design using a data-driven continuous representation of molecules. arXiv preprint arXiv:1610.02415.





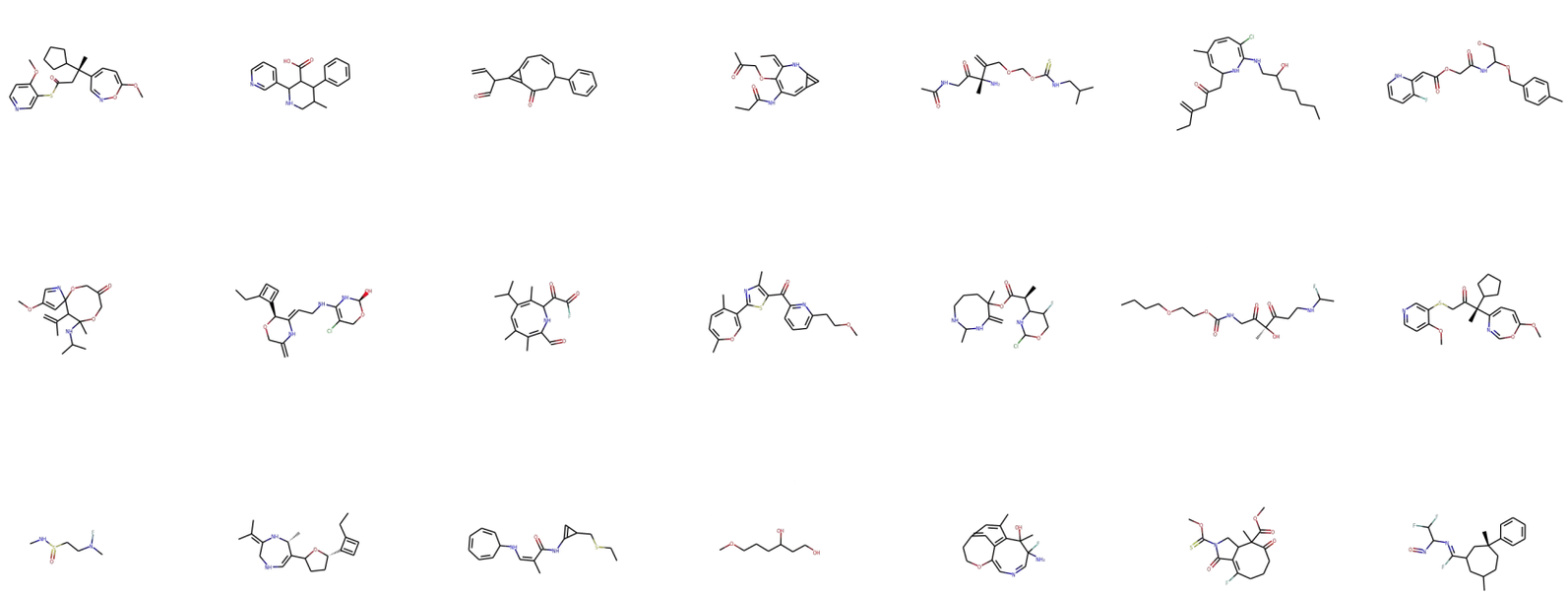
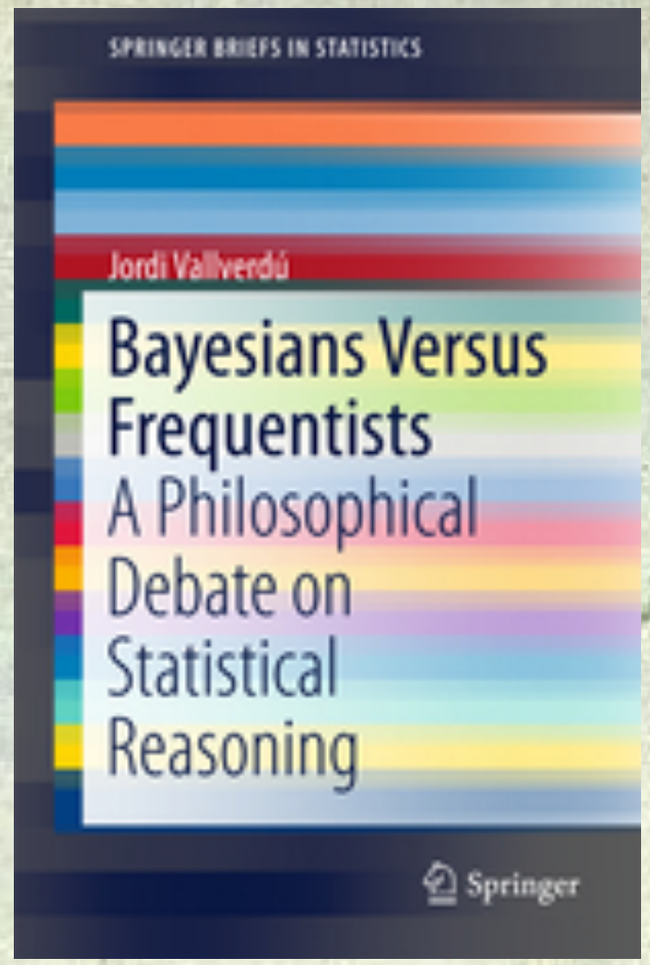
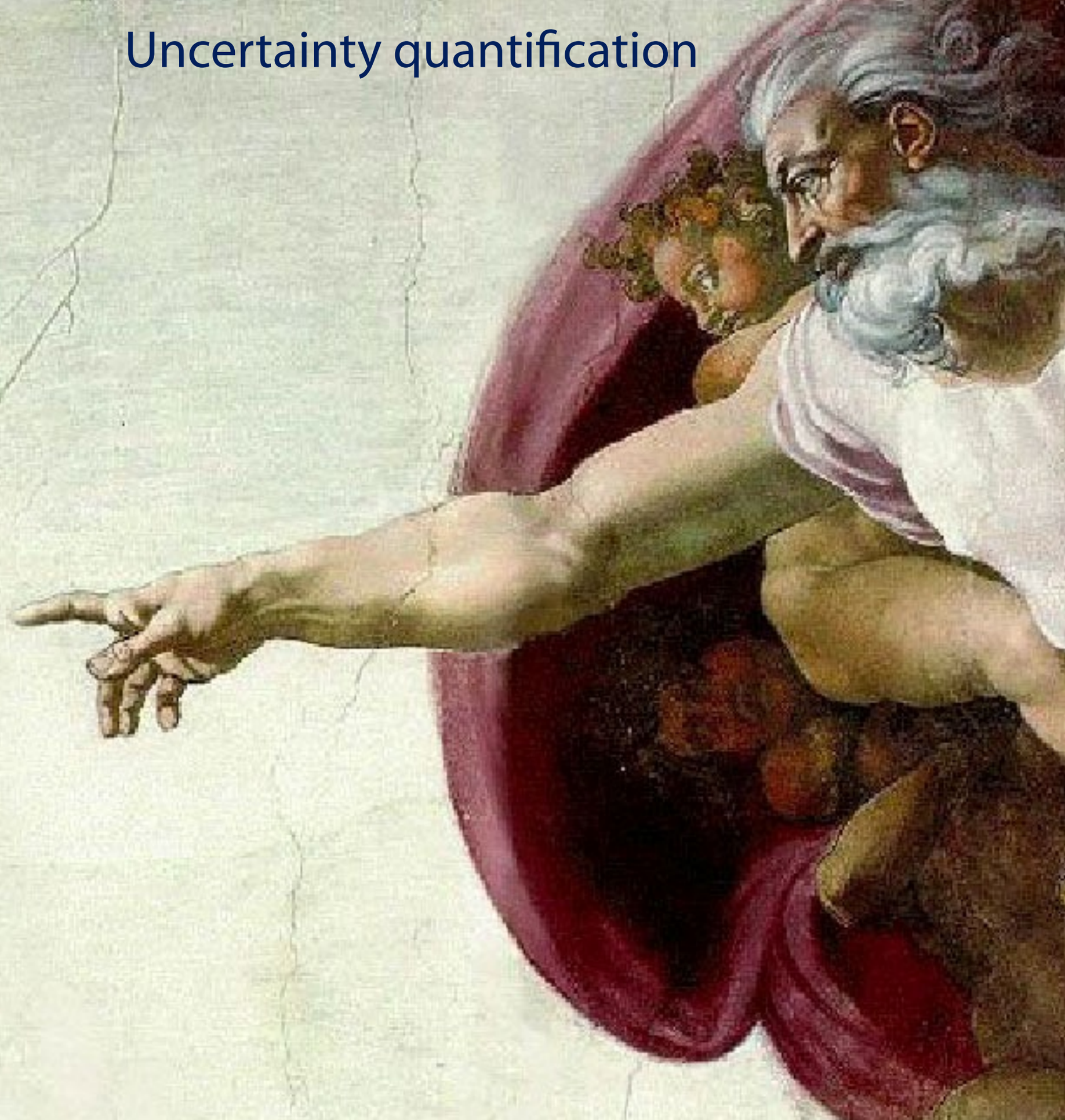


Figure 8: Molecules decoded from randomly-sampled points in the latent space of the ZINC VAE.

Uncertainty quantification





Course goals

- Introduce a set of enabling machine learning tools for computational science and engineering applications.
- Utilize prior knowledge and modeling expertise to construct more structured and efficient predictive algorithms.
- Introduce a new type of scientific computing in which machine learning tools are used to develop new solvers, discover new models, perform sensitivity analysis/uncertainty quantification, guide the acquisition of new information, and solve design and optimization problems.

Course logistics

- Duration: 14 Weeks
- Time: Tuesdays and Thursdays, 4:30 pm to 6:00 pm
- Dates: January 10 to April 25, 2018. No classes on Tuesday March 6, and Thursday March 8.
- Location: Towne Building, Room 309
- Office Hours: Thursdays 11:00 am to 1:00 pm at Office 359C (3401 Walnut Street).

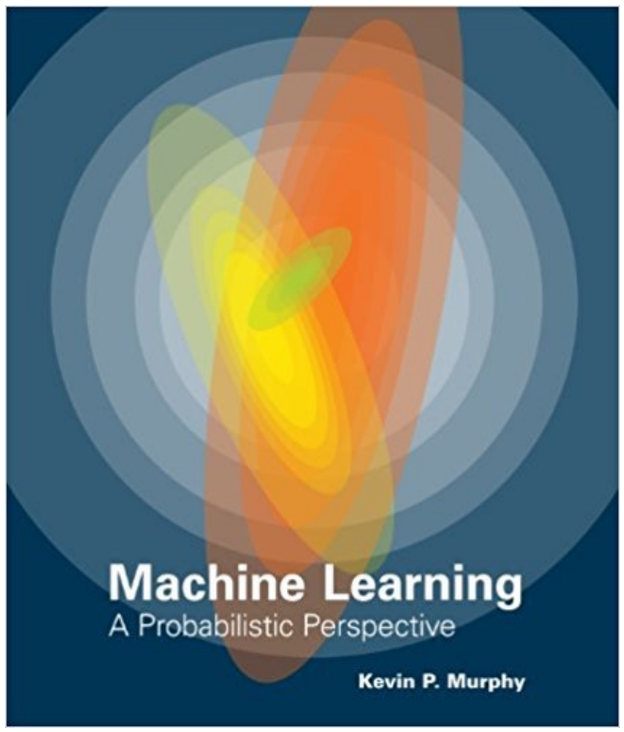
For more details visit the course website:
https://www.seas.upenn.edu/~enm540/

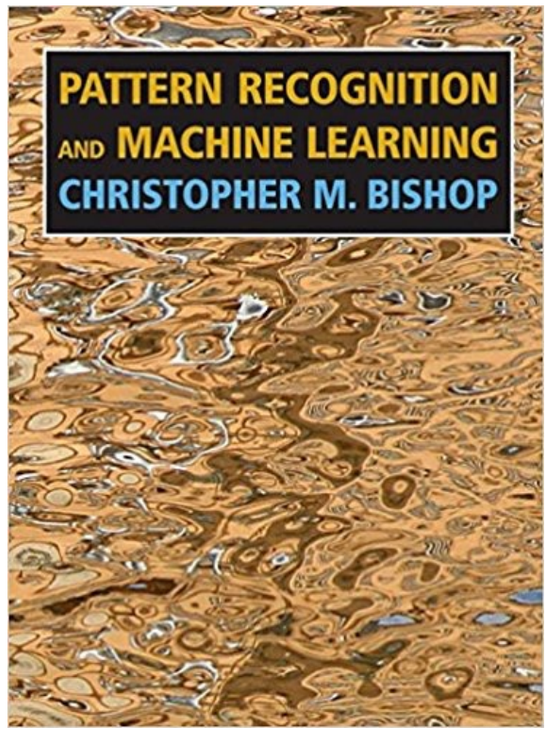
Slides and code can be found on Github:
https://github.com/PredictiveIntelligenceLab/ENM-540

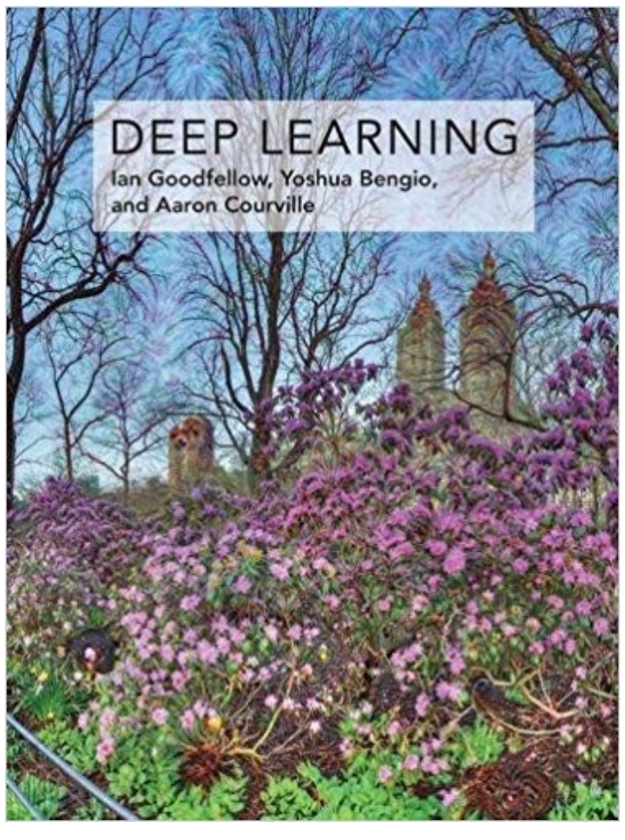
Course outline

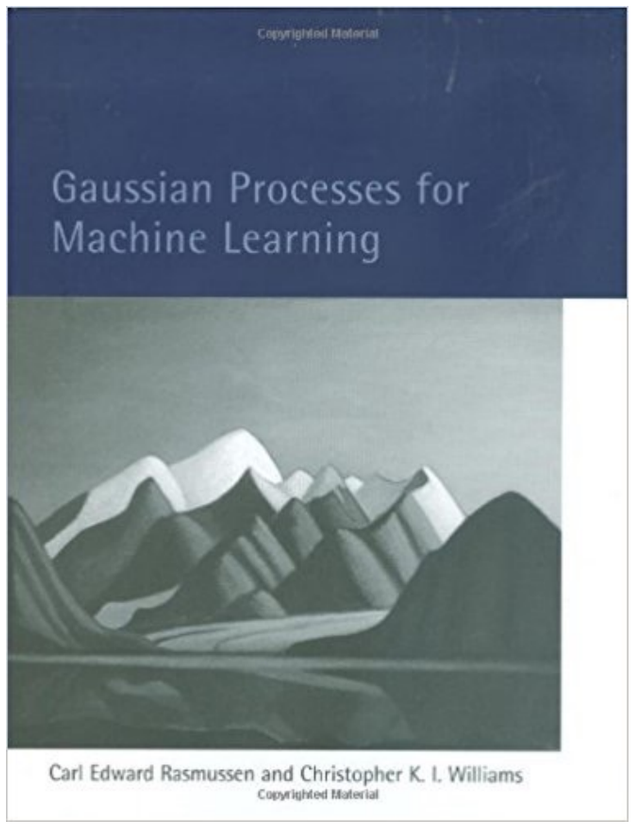
- Week 1: Introduction to data-driven modeling, motivating examples and course outline.
- Week 2: Probability theory primer through the lens of Bayesian linear regression.
- Week 3: Optimization: gradients and Hessians, gradient descent, Newton's algorithm, stochastic gradient descent.
- Week 4: Deep neural networks: back-propagation, over-fitting and regularization, automatic differentiation.
- Week 5: Image classification with convolutional neural networks.
- Week 6: Recurrent neural networks and LSTMs.
- Week 7: Sampling and quantification of uncertainty.
- Week 8: Gaussian processes, multi-output Gaussian processes and multi-fidelity modeling.
- Week 9: Bayesian optimization and active learning.
- Week 10: Probabilistic scientific computing: Gaussian process regression meets differential equations.
- Week 11: Unsupervised learning: principal component analysis, Gaussian process latent variable models.
- Week 12: Variational auto-encoders and conditional variational auto-encoders.
- Week 13: Generative adversarial networks.
- Week 14: Class presentations of final papers.

Textbooks


Machine Learning
A Probabilistic Perspective
Kevin P. Murphy


PATTERN RECOGNITION
AND MACHINE LEARNING
CHRISTOPHER M. BISHOP


DEEP LEARNING
Ian Goodfellow, Yoshua Bengio,
and Aaron Courville


Copyrighted Material
Gaussian Processes for
Machine Learning
Carl Edward Rasmussen and Christopher K. I. Williams
Copyrighted Material

Software

Week 1 - Setting up your computing enviroment

Consult the Web for instructions and make sure you have the following software properly installed and working on your laptop:

- MATLAB: Free academic license and installation instructions available through CETS.
- Python 2.7 or higher, set up for scientific computing. The simplest way to set this up is by installing the Anaconda distribution.
- Tensorflow. If you have access to GPU hardware, make sure you install a version with GPU support.
- PyTorch. If you have access to GPU hardware, make sure you install a version with GPU support.
- Autograd. Efficiently computes derivatives of numpy code.
- Jupyter notebook. You will need this in order to follow some of the in-class tutorials.
- Git. You will need this in order to download and stay in sync with the latest code we will develop in class.

Make sure you can succesfully run tutorials provided in the course Github page:

- Linear regression in Numpy
- Linear regression in Numpy with Autograd
- Linear regression in PyTorch
- Linear regression in Tensorflow
- Interactive Jupyter notebook