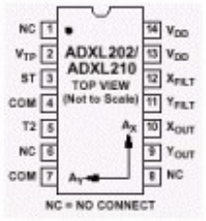
ENM 531: Data-driven Modeling and Probabilistic Scientific Computing

Lecture #0: Introduction, motivation and course logistics

Paris Perdikaris
January 12, 2023

Roadmap to Trillion Sensors



Accelerometer


Gyro


Pendulum Resistive
Tilt Sensors


Piezo Bend Sensor

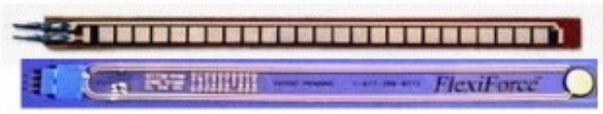

Metal Detector

Gas Sensor


Gieger-Muller
Radiation Sensor


Digital Infrared Ranging


CDS Cell
Resistive Light Sensor



Resistive Bend Sensors

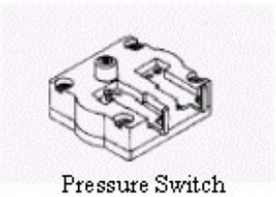

UV Detector

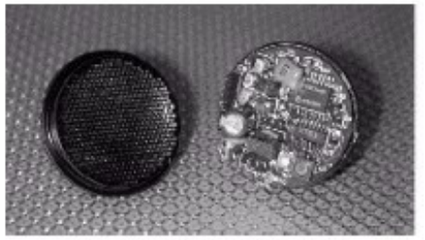

Pyroelectric Detector

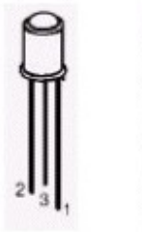

Limit Switch

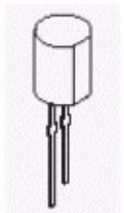

Mechanical Tilt Sensors

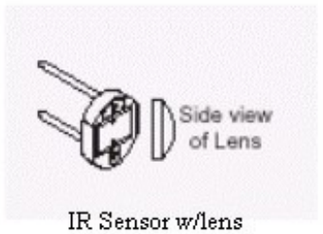

Touch Switch


Pressure Switch


Miniature Polaroid Sensor

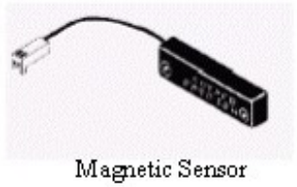



IR Pin
Diode

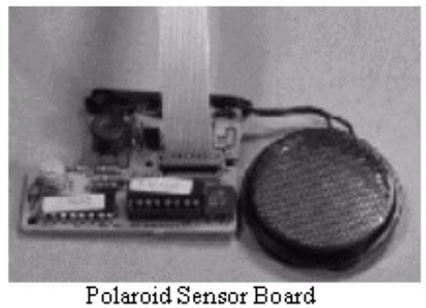


IR Sensor w/lens


Thyristor


Magnetic Sensor


Hall Effect
Magnetic Field
Sensors

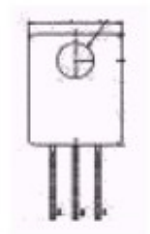

Polaroid Sensor Board

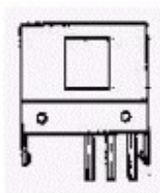

IR Reflection
Sensor


IR Amplifier Sensor


IRDA Transceiver


Magnetic Reed Switch

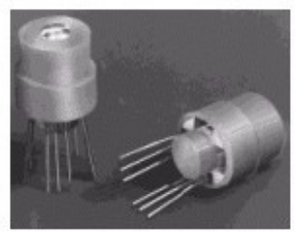

Lite-On IR
Remote Receiver

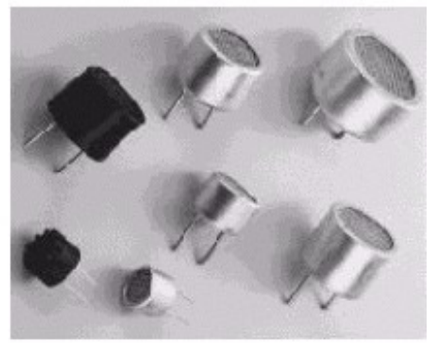

Radio Shack
Remote Receiver


IR Modulator
Receiver


Solar Cell


Compass

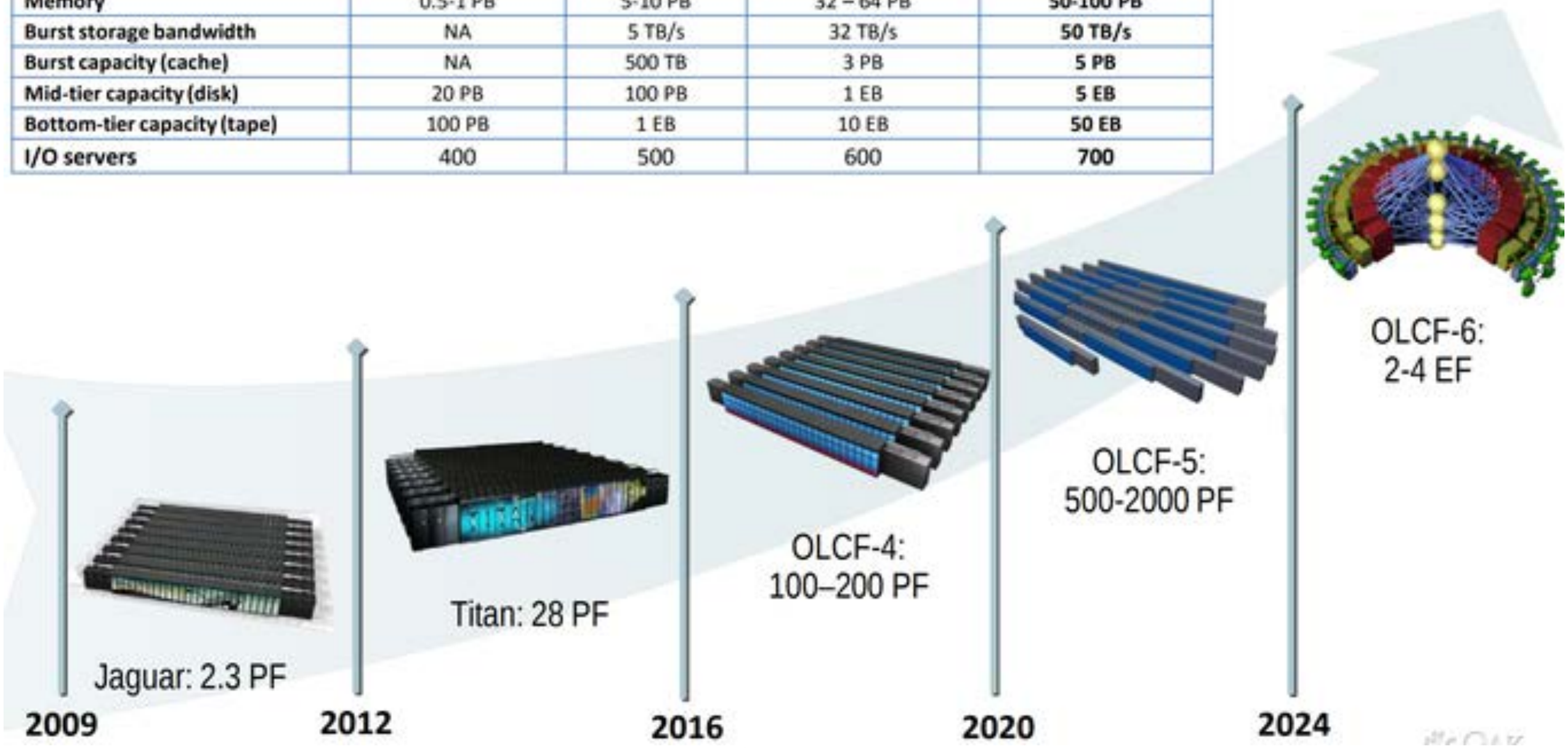

Compass


Piezo Ultrasonic Transducers

Sensors are expected to generate a bronto-bytes (1,000 trillion trillion) of data by 2025!

Roadmap to Exascale computing

	2012	2016	2020	2024
Peak flops	10-20 PF	100-200 PF	500-2000 PF	**2000 - 4000 PF**
Memory	0.5-1 PB	5-10 PB	32 – 64 PB	**50-100 PB**
Burst storage bandwidth	NA	5 TB/s	32 TB/s	**50 TB/s**
Burst capacity (cache)	NA	500 TB	3 PB	**5 PB**
Mid-tier capacity (disk)	20 PB	100 PB	1 EB	**5 EB**
Bottom-tier capacity (tape)	100 PB	1 EB	10 EB	**50 EB**
I/O servers	400	500	600	**700**

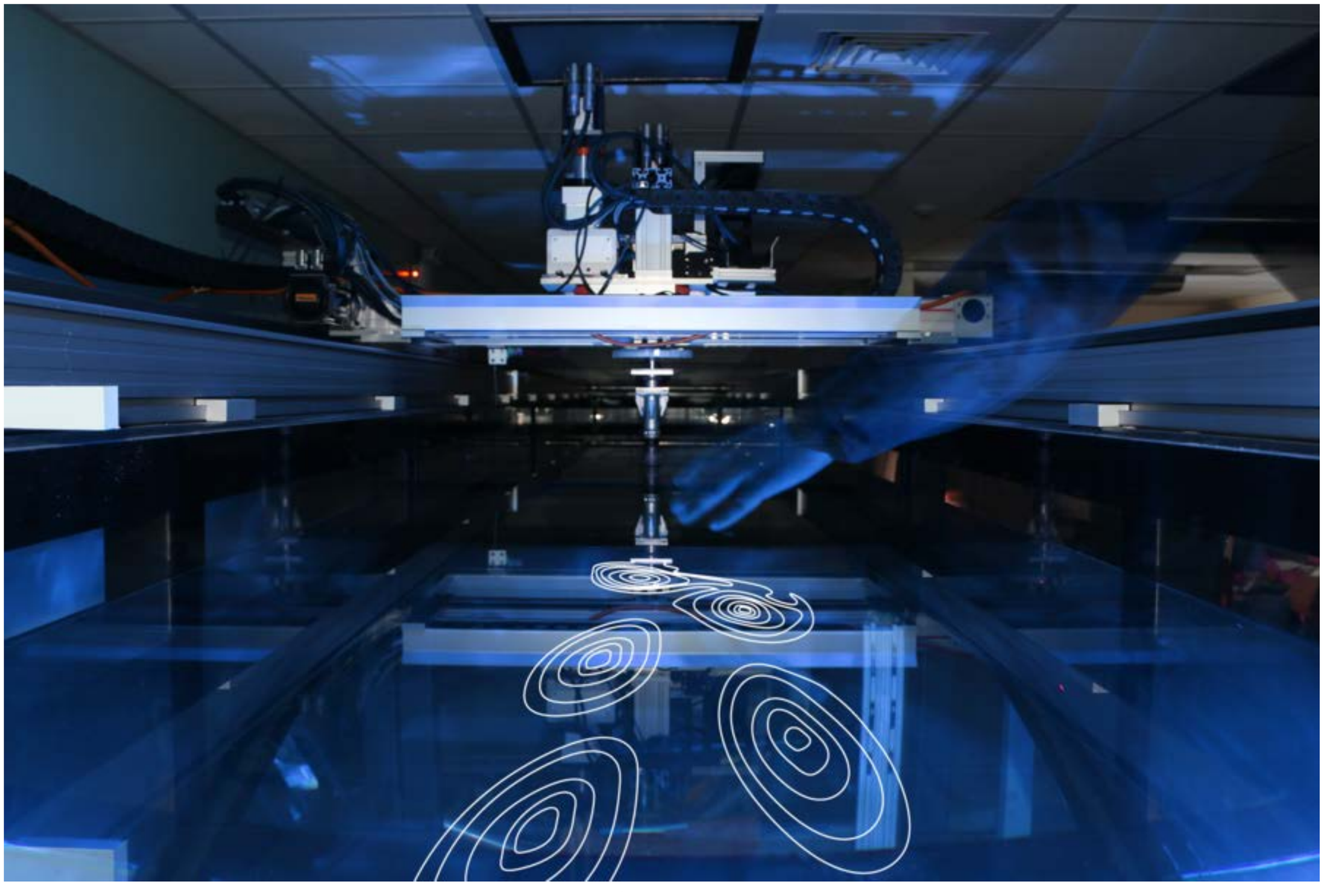


1 ExaFlop: 10^{18} floating point operations per second.

Autonomy

Tesla AI Day: https://www.youtube.com/watch?v=ODSJsviD_SU

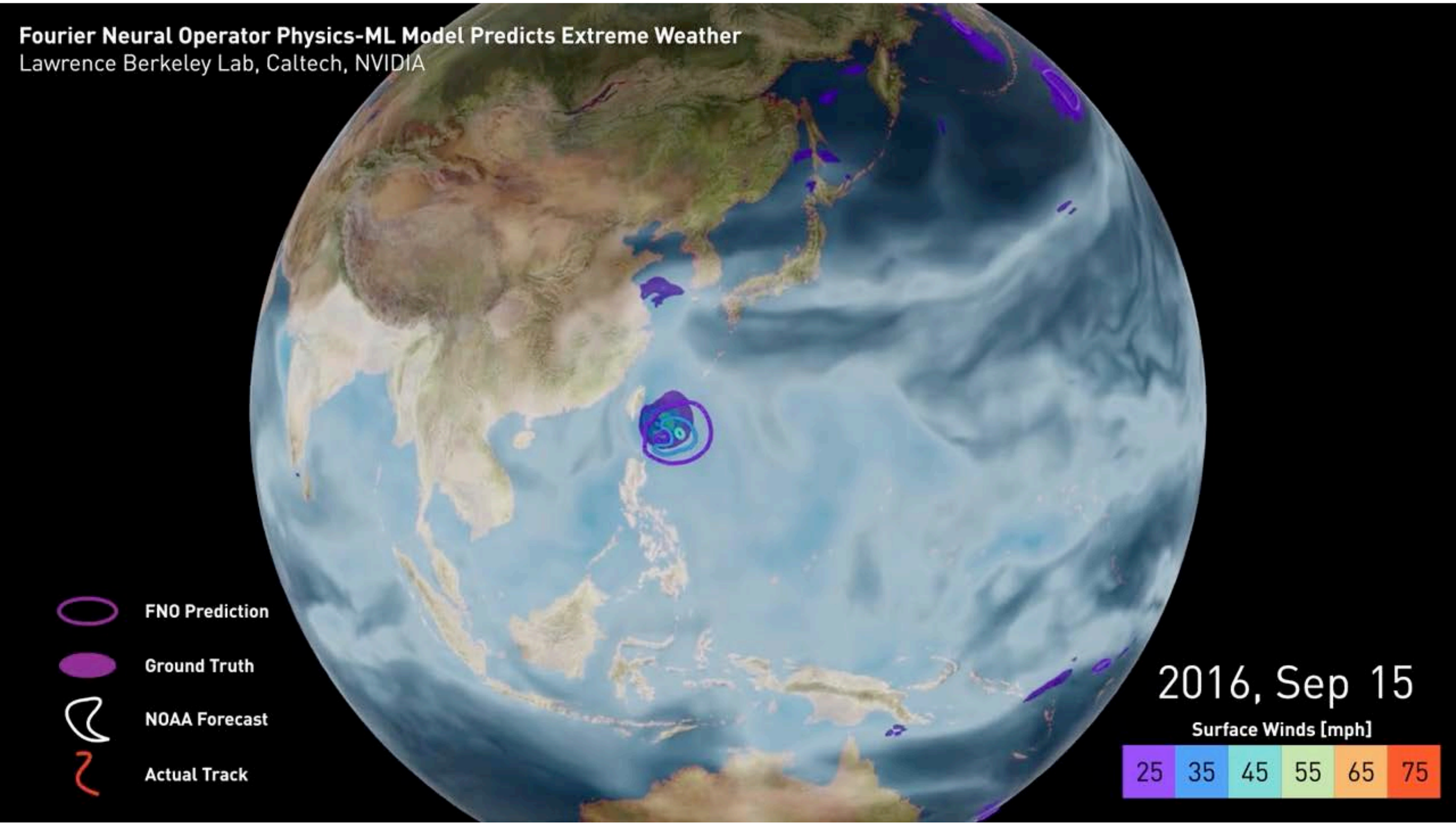
Intelligent experimentation



"In its first year of operation, the Intelligent Towing Tank (ITT) conducted about 100,000 total experiments, essentially completing the equivalent of a PhD student's five years' worth of experiments in a matter of weeks."

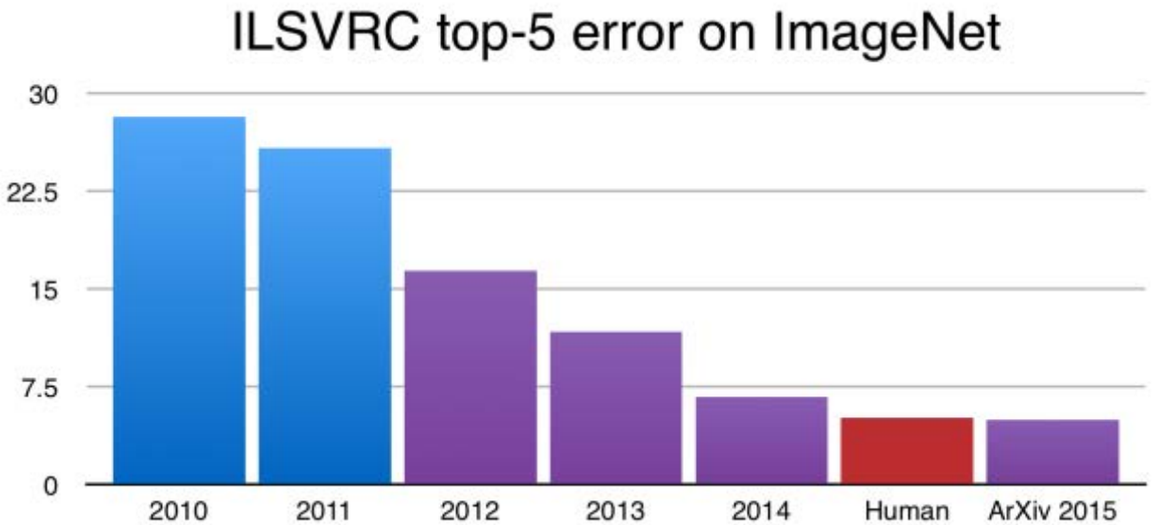
https://news.mit.edu/2019/intelligent-towing-tank-propels-research-1209

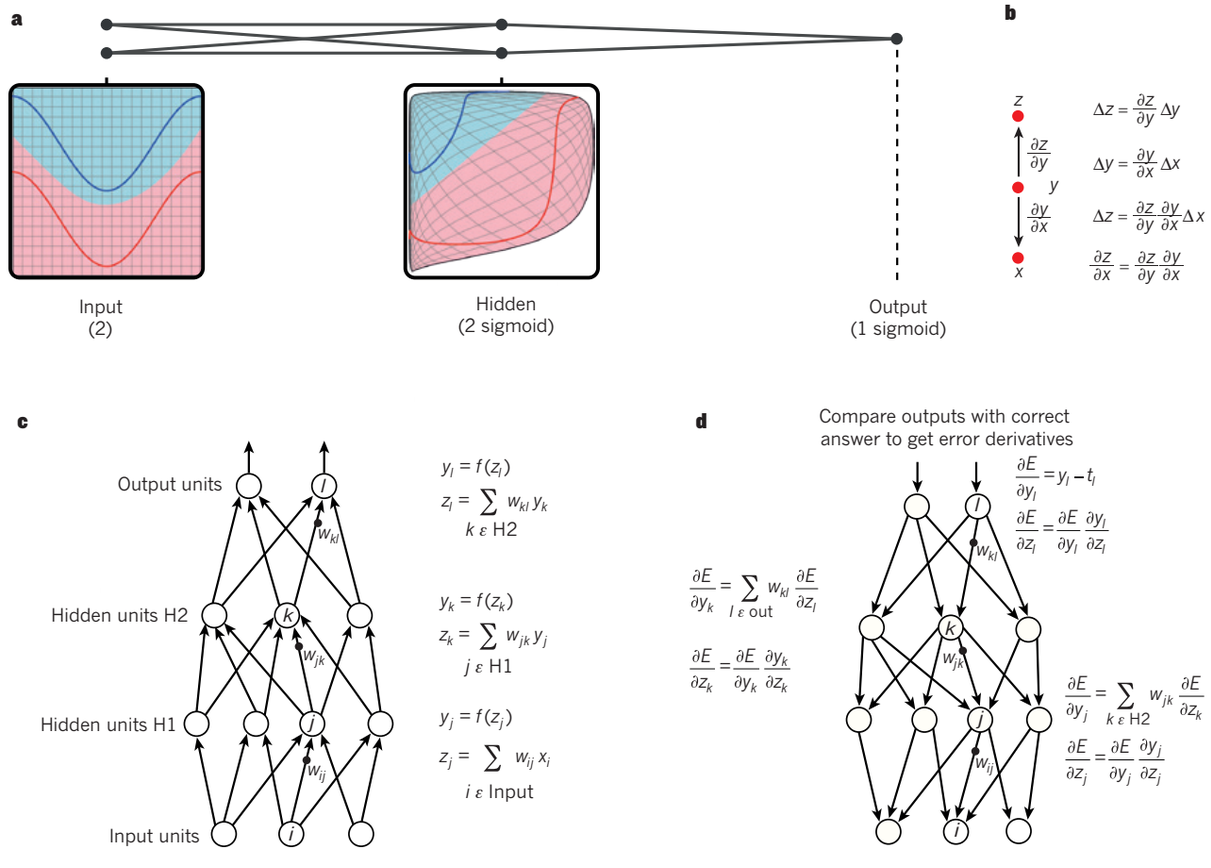
Predictive science and engineering



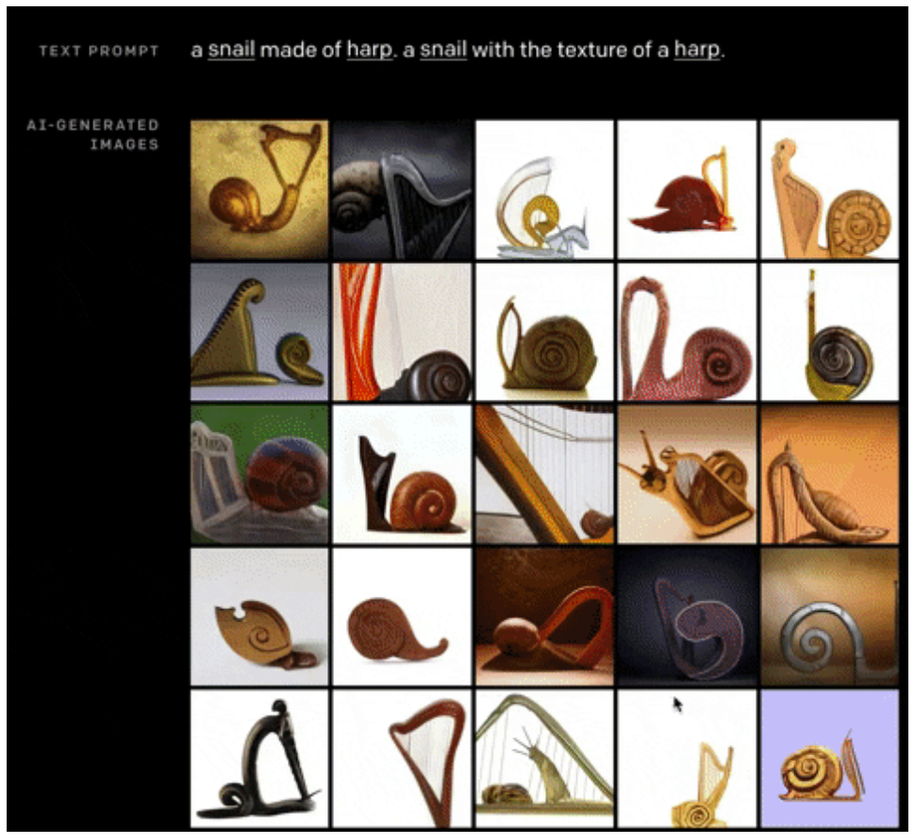
https://blogs.nvidia.com/blog/2021/11/16/ai-science-climate-change/

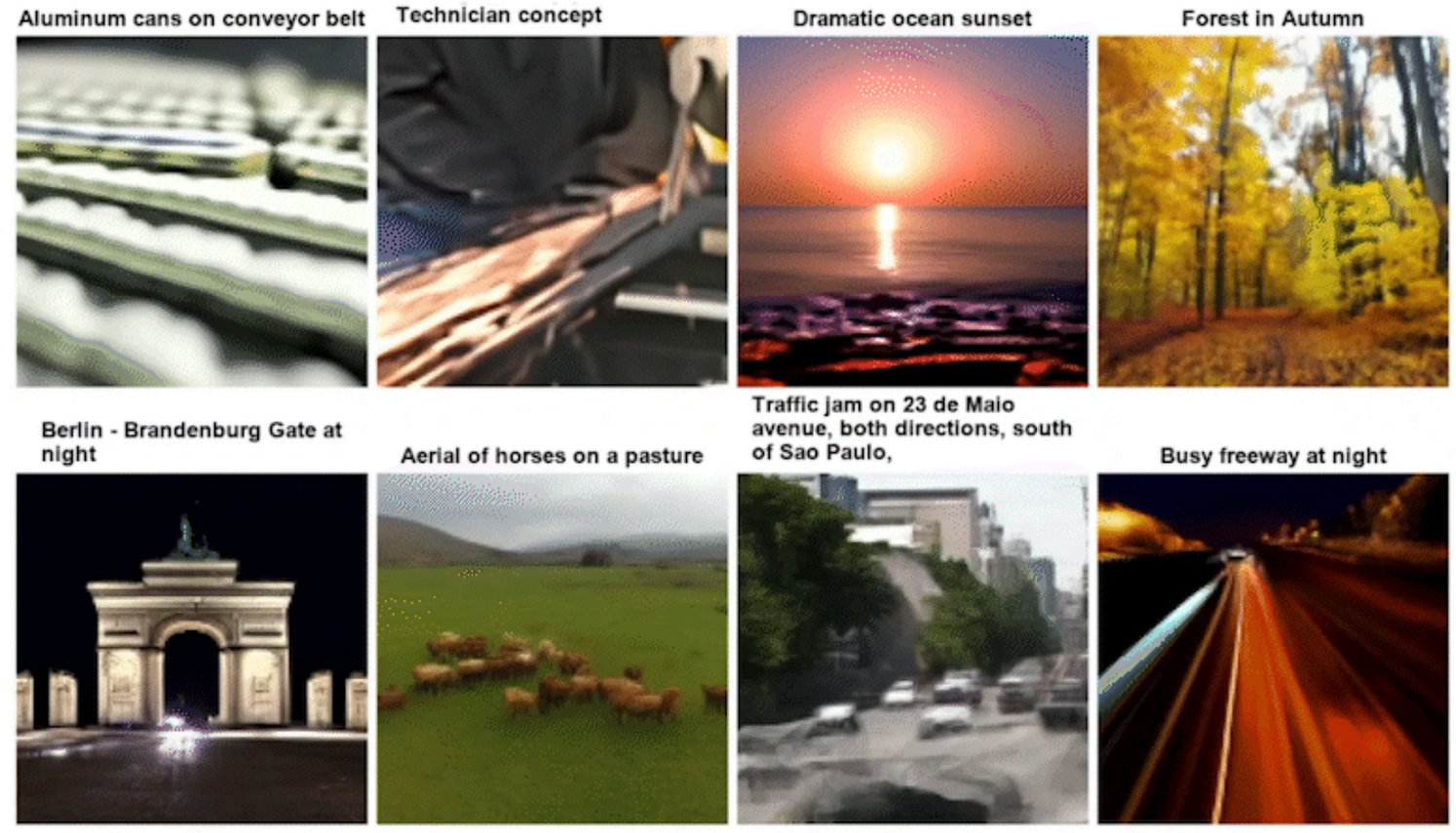
Recent success of machine learning

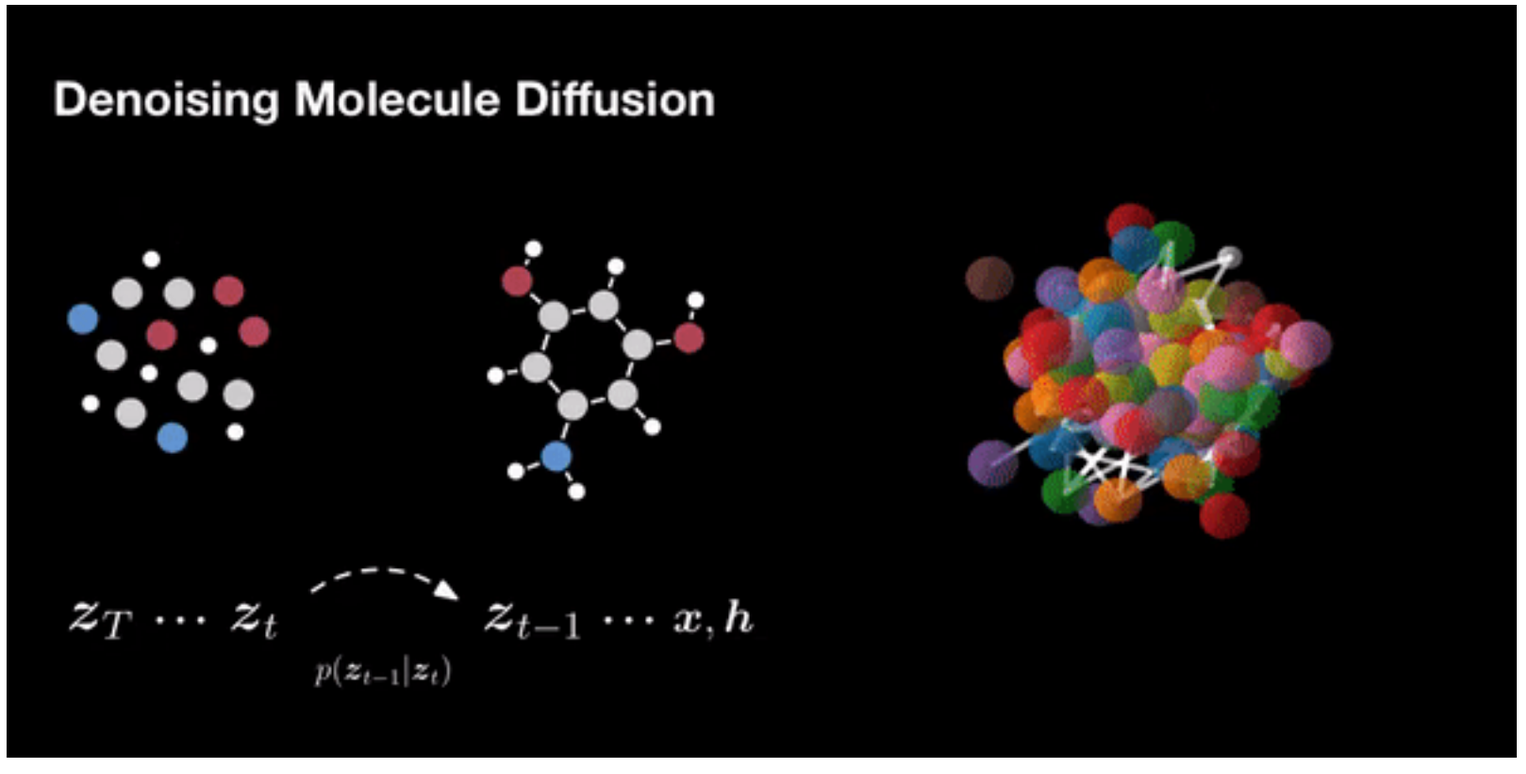




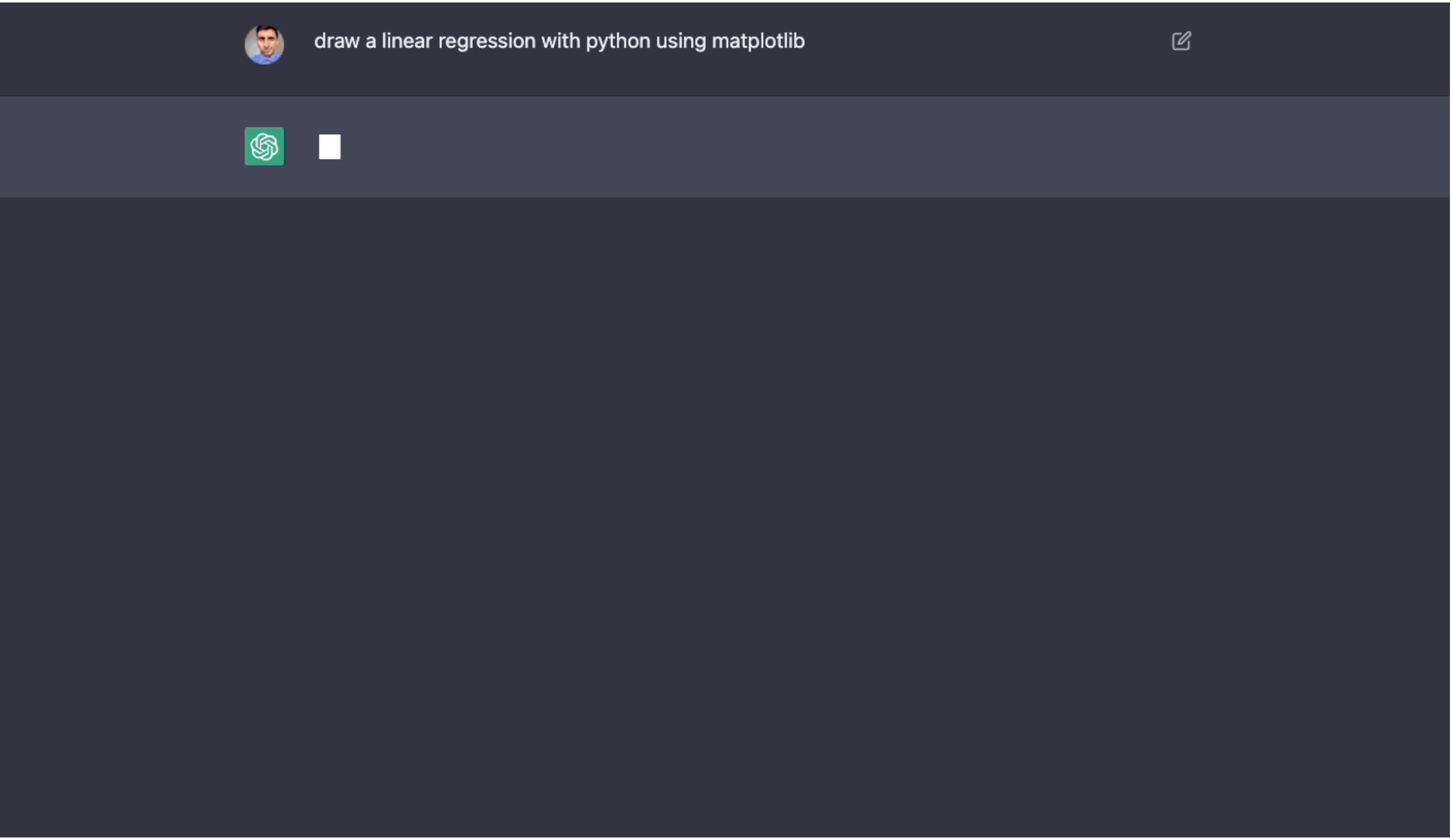
Working with high-dimensional data







Working with multiple data modalities

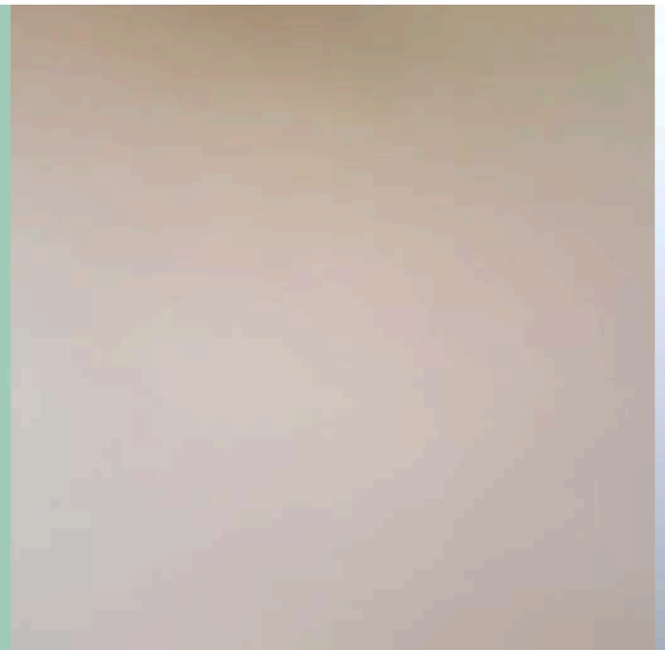


Data-driven science & engineering

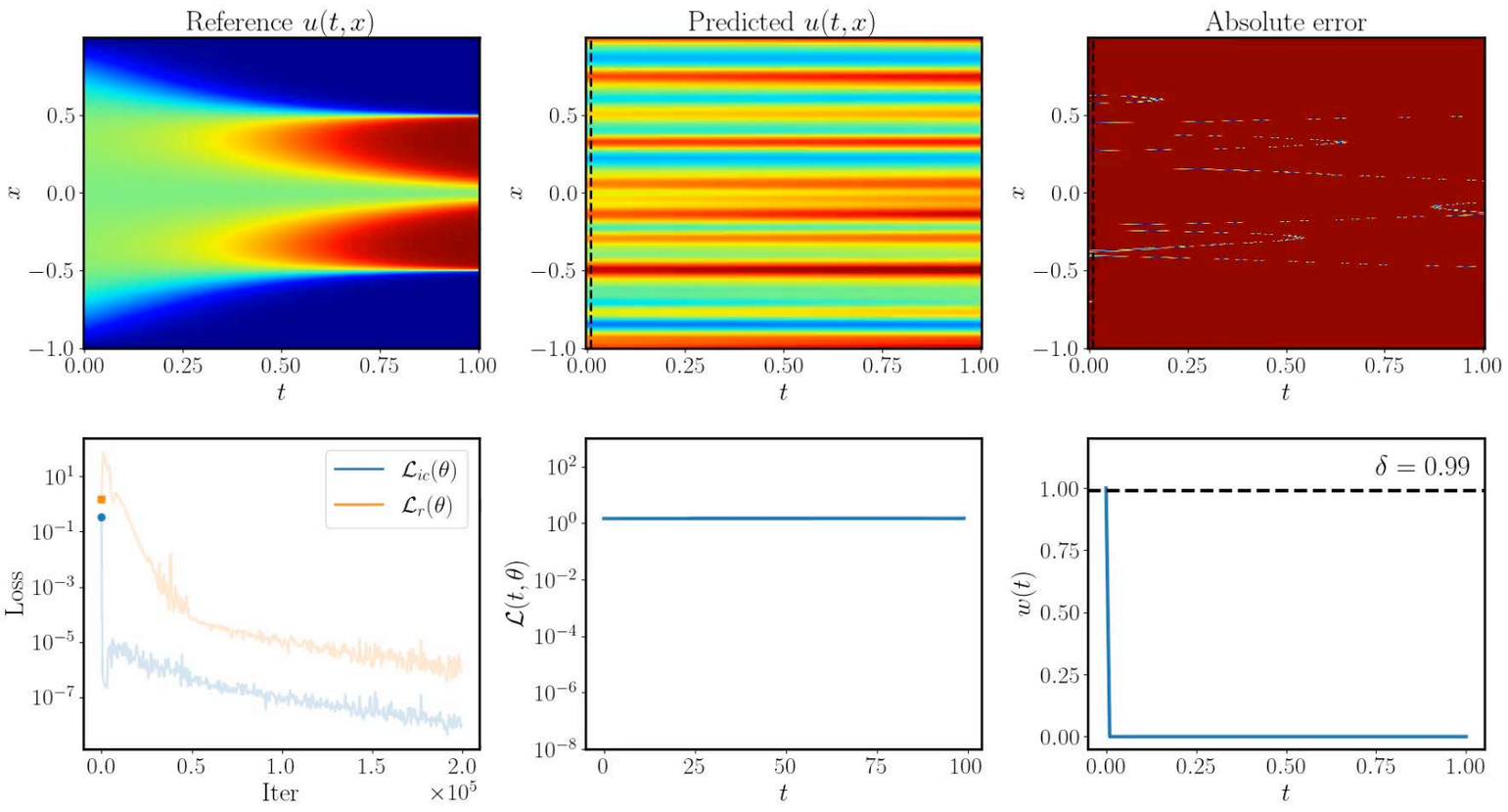
Neural gigapixel images

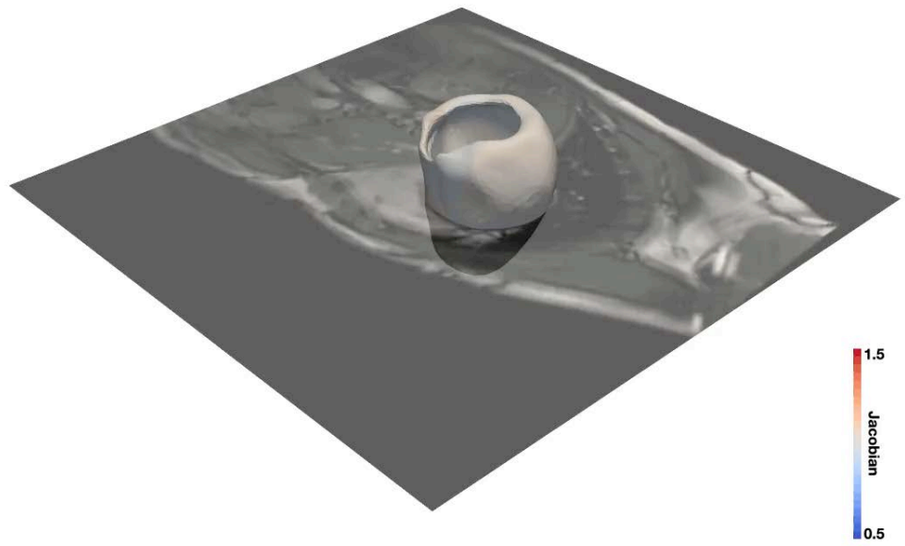
Neural SDF

NeRF

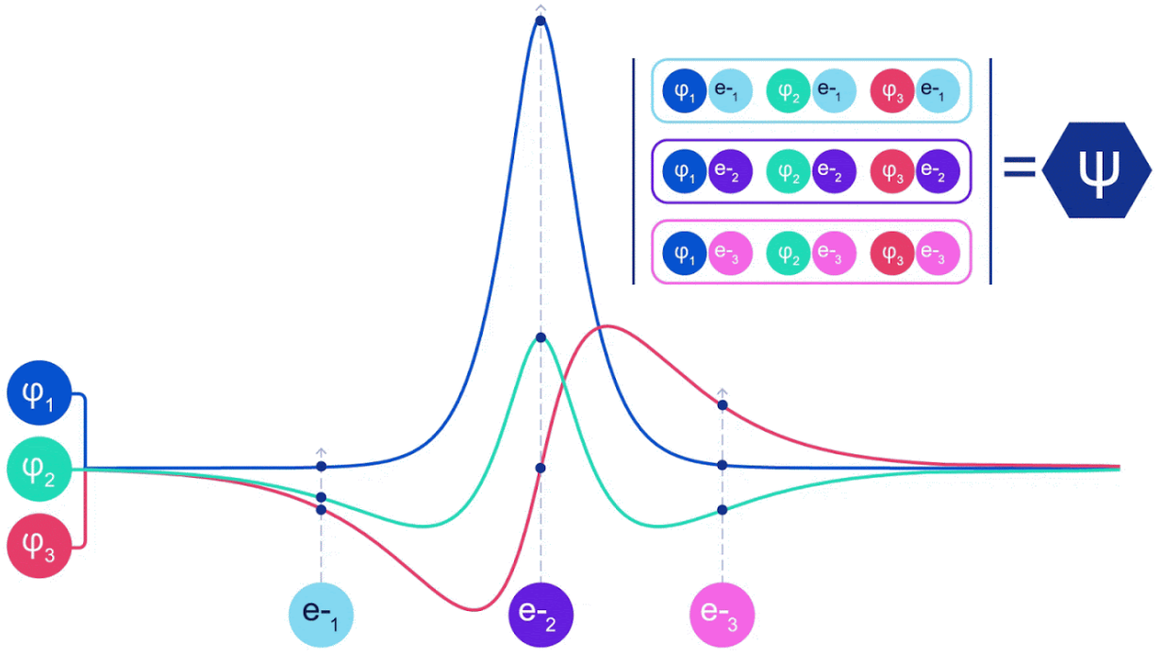


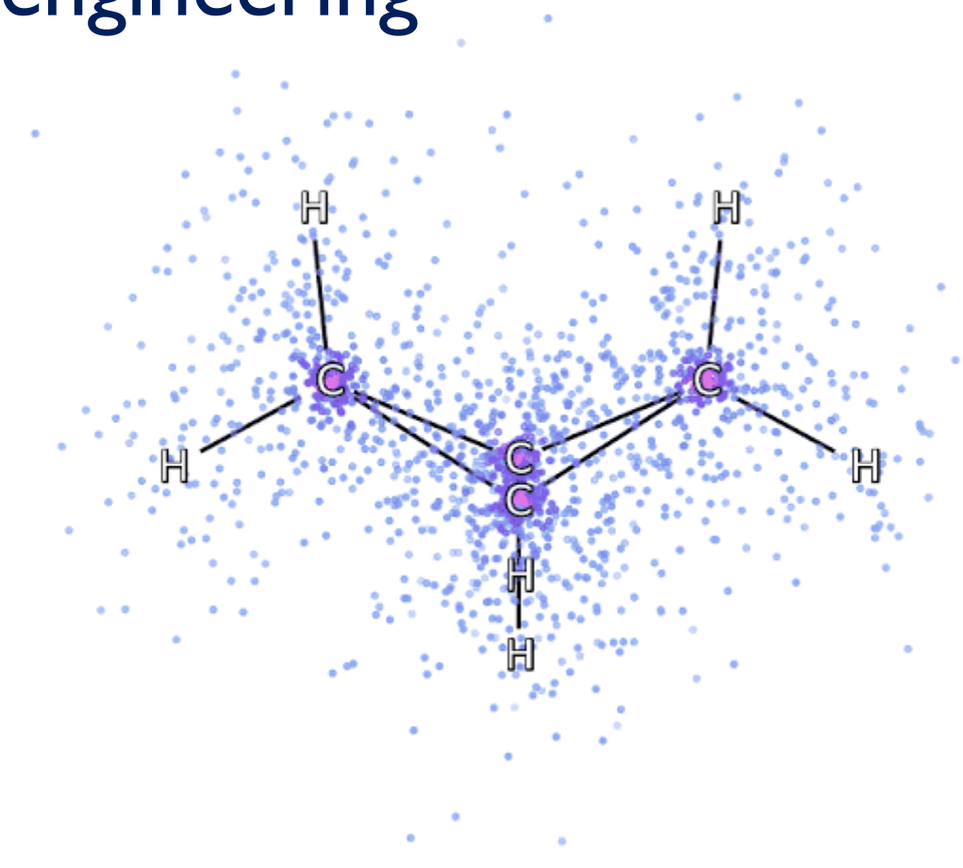
Neural volume

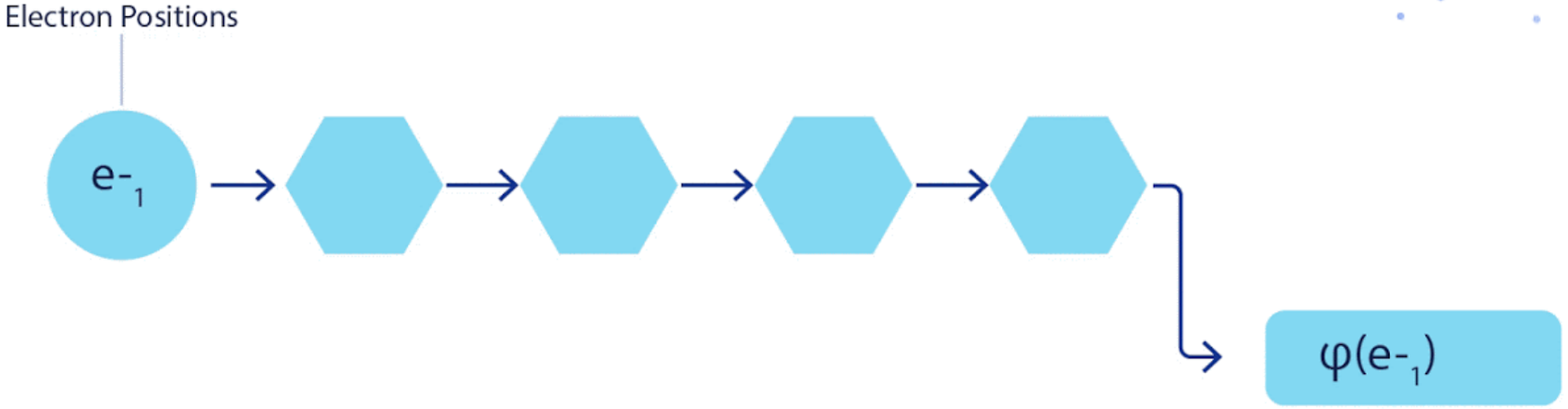




Data-driven science & engineering

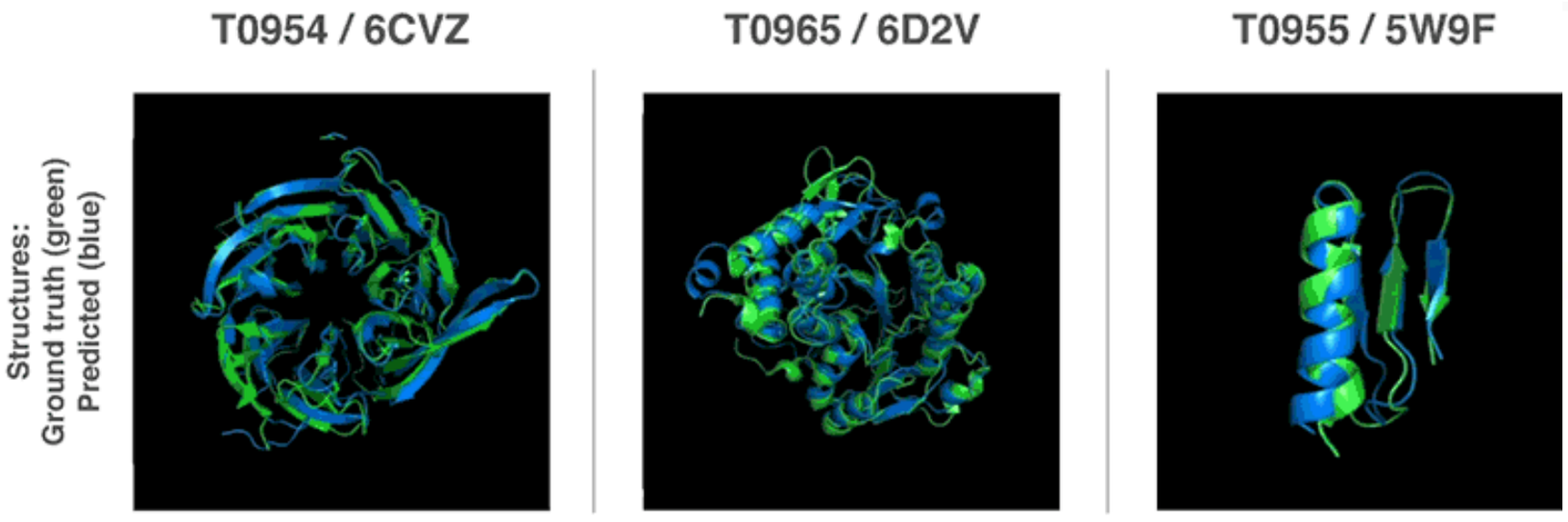


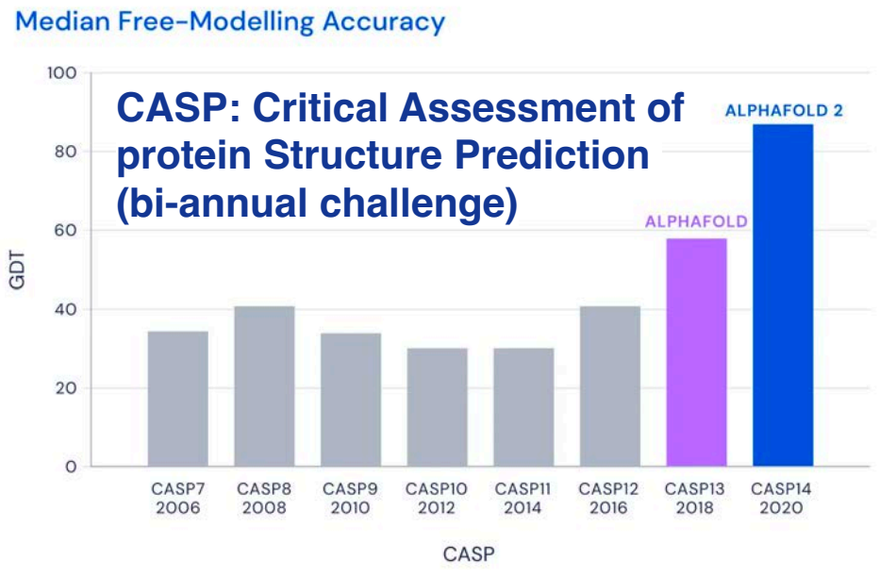


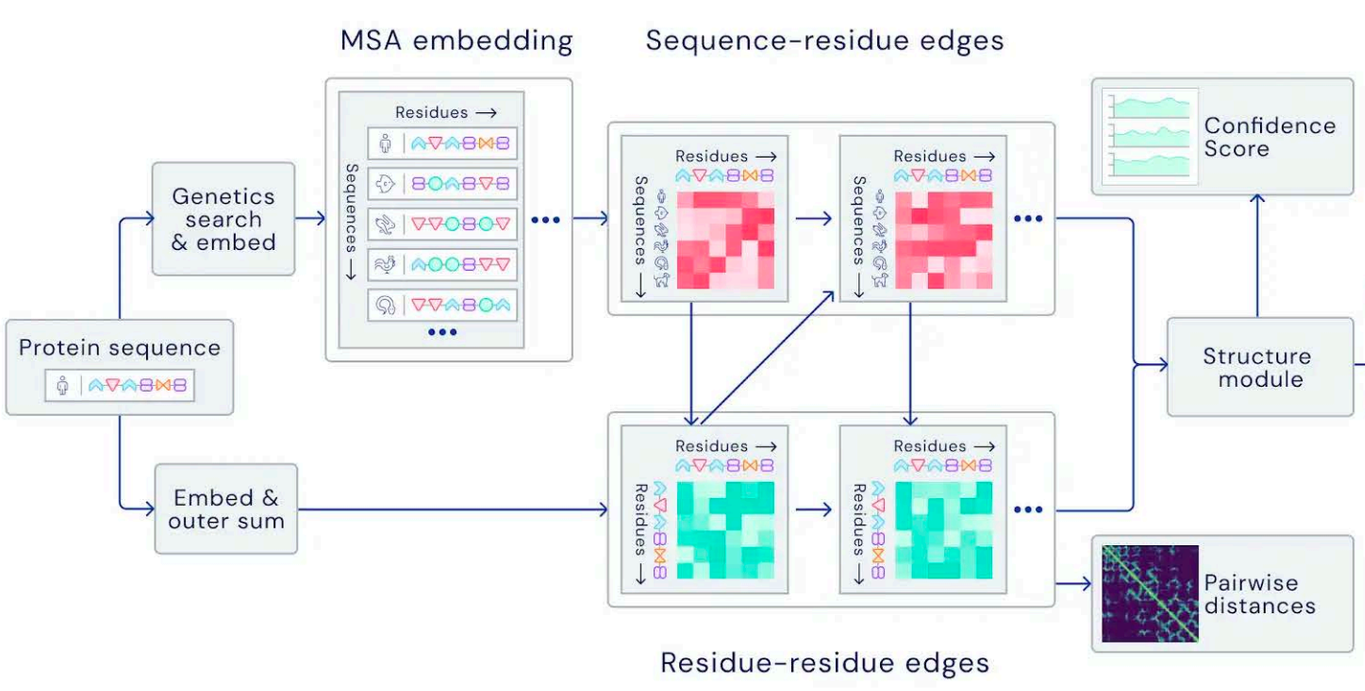


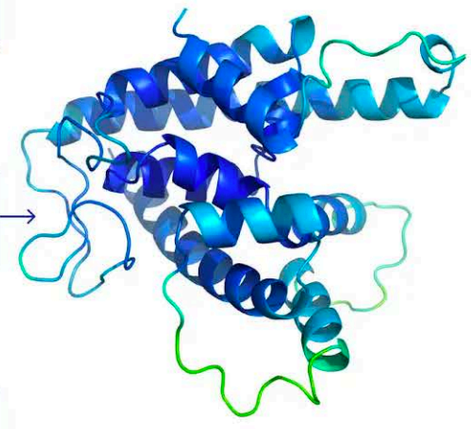
- [FermiNet: Quantum Physics and Chemistry from First Principles] https://deepmind.com/blog/article/FermiNet

Data-driven science & engineering





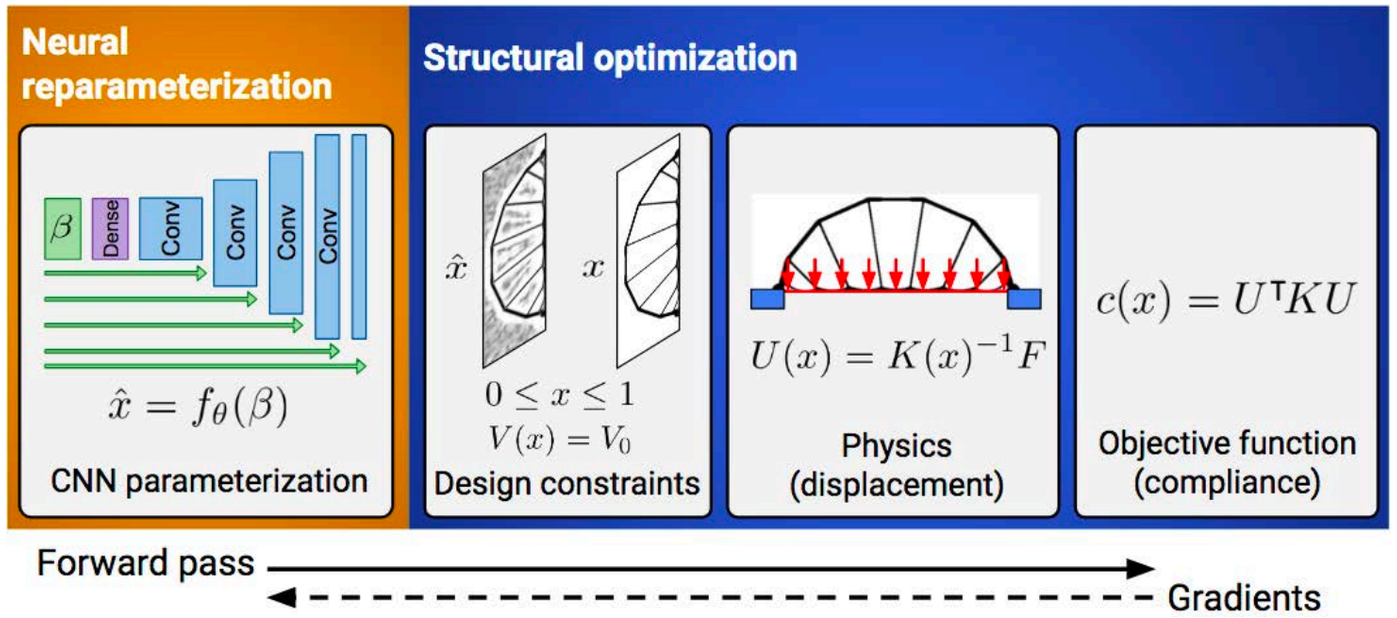


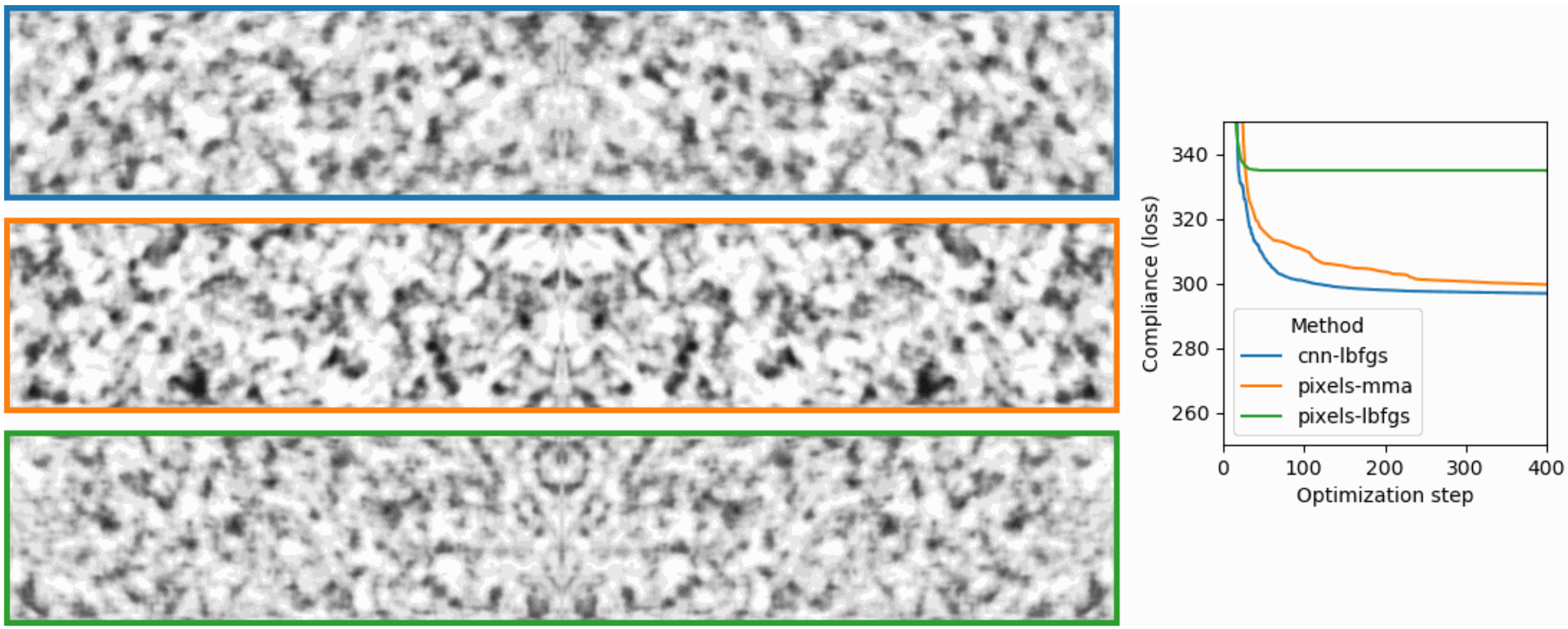


AplhaFold2: A step towards tackling the protein folding problem by training semi-supervised deep learning models on publicly available data consisting of ~170,000 protein structures from the protein data bank together with large databases containing protein sequences of unknown structure.

https://deepmind.com/blog/article/alphafold-a-solution-to-a-50-year-old-grand-challenge-in-biology

Data-driven science & engineering





Optimizing a bridge structure. In the top frame, optimization happens in the weight space of a CNN. In the next two frames it happens on a finite element grid.

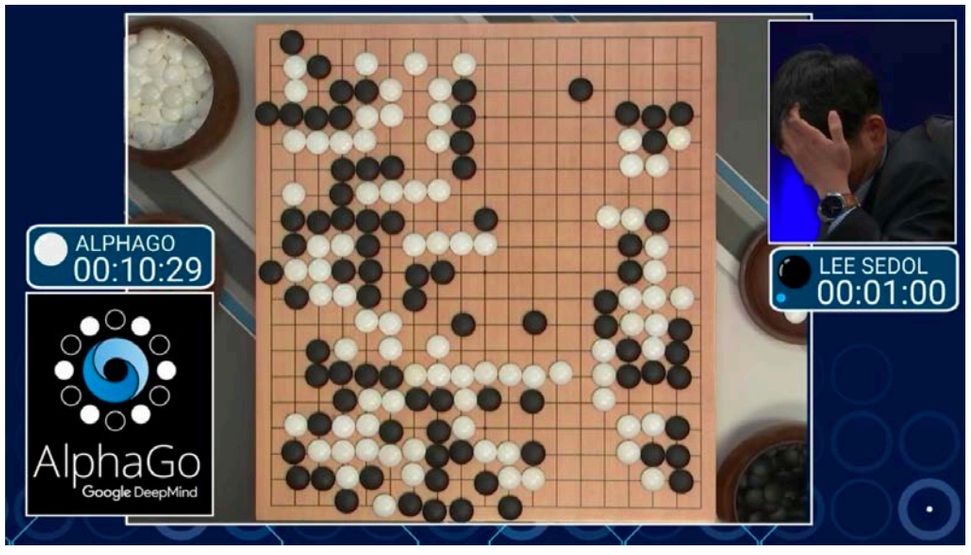
https://greydanus.github.io/2019/12/15/neural-reparam/

From predictions to decisions

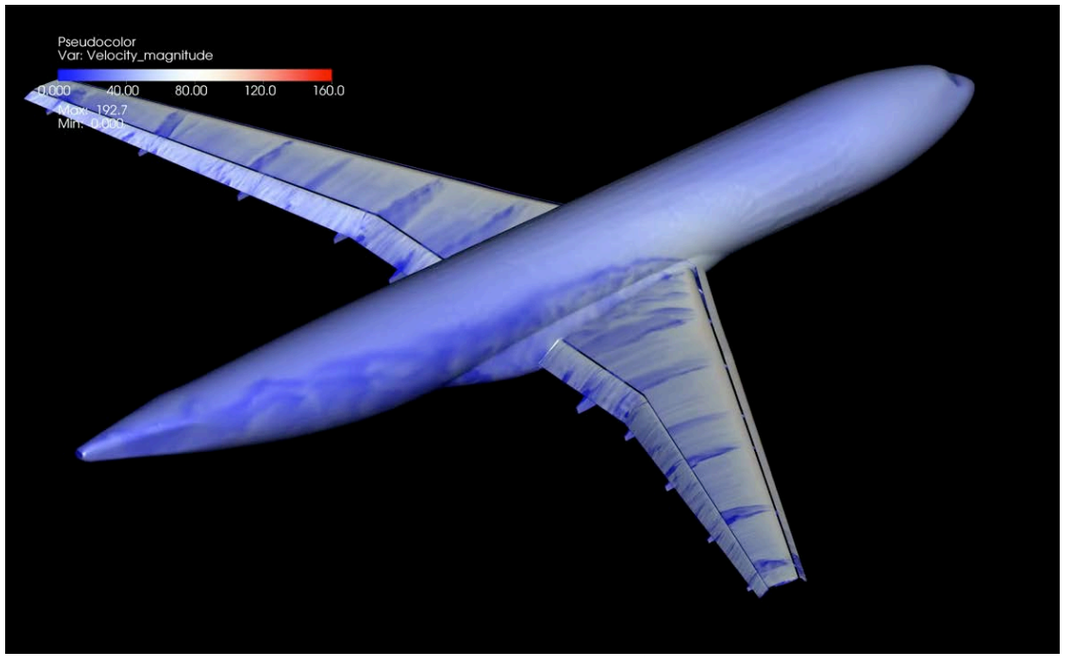
Google DeepMind's Deep Q-learning

The algorithm will play Atari breakout.

The most important thing to know is that all the agent is given is sensory input (what you see on the screen) and it was ordered to maximize the score on the screen.



Data- vs Model-driven science



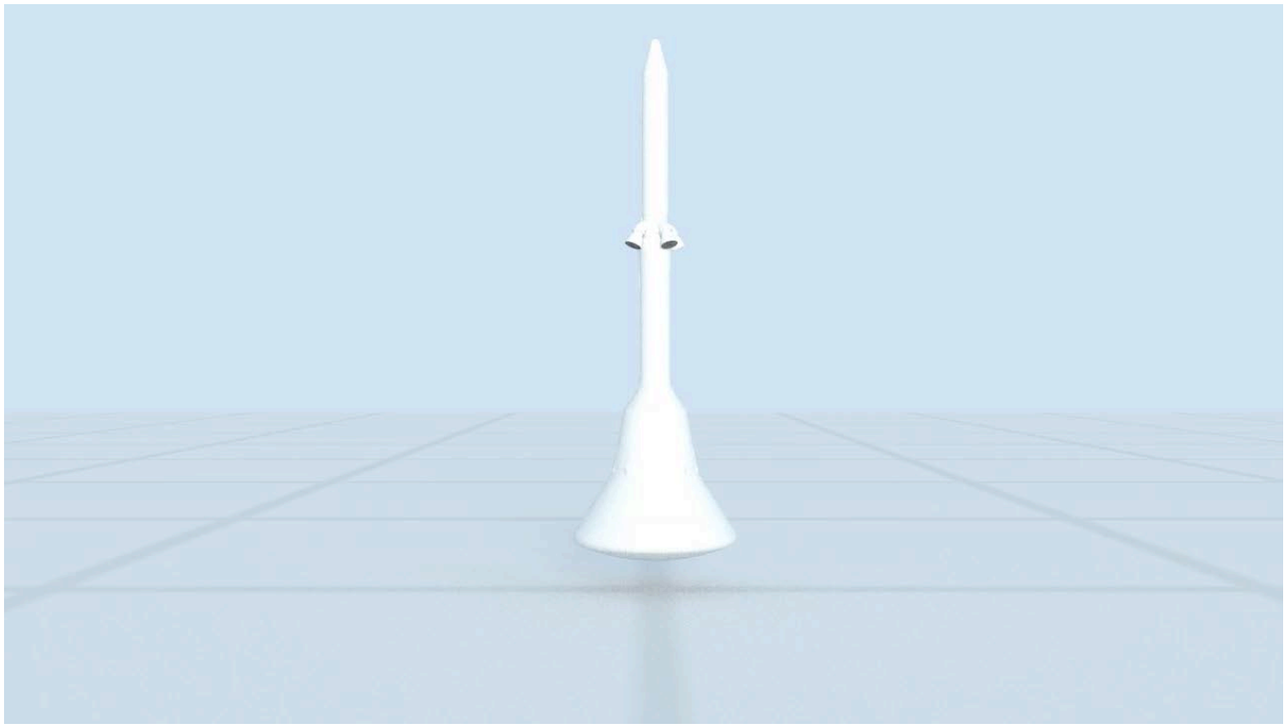
Data — *Models*

Machine learning

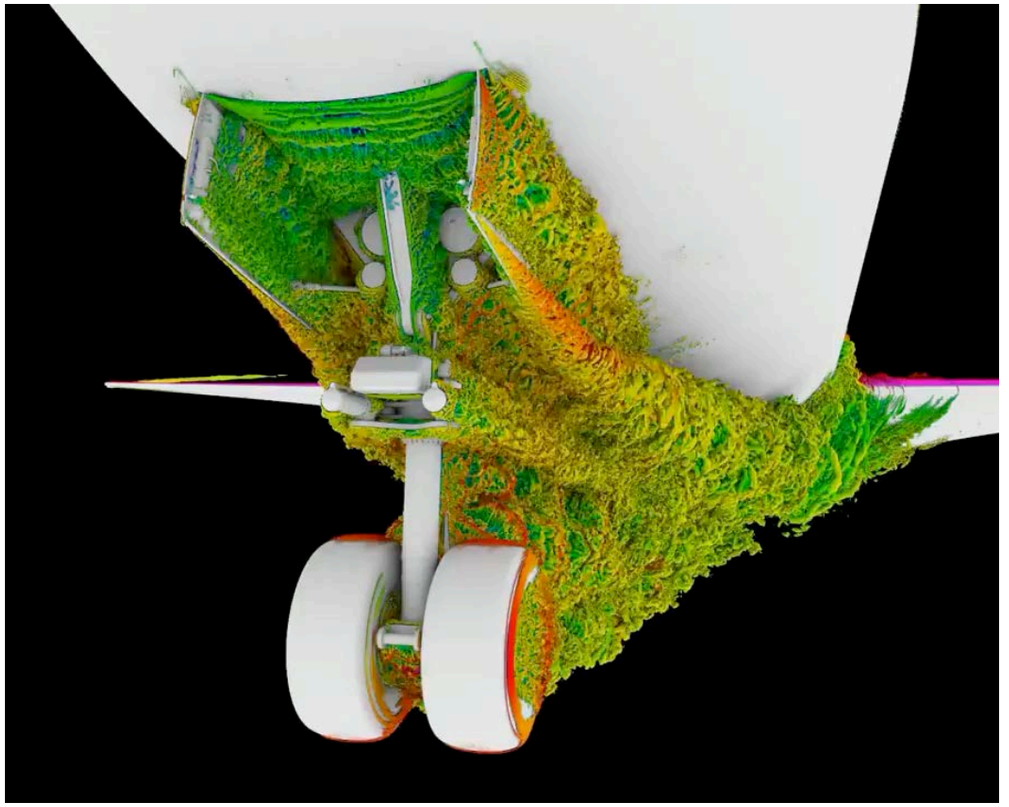
CSE

Computational science & engineering

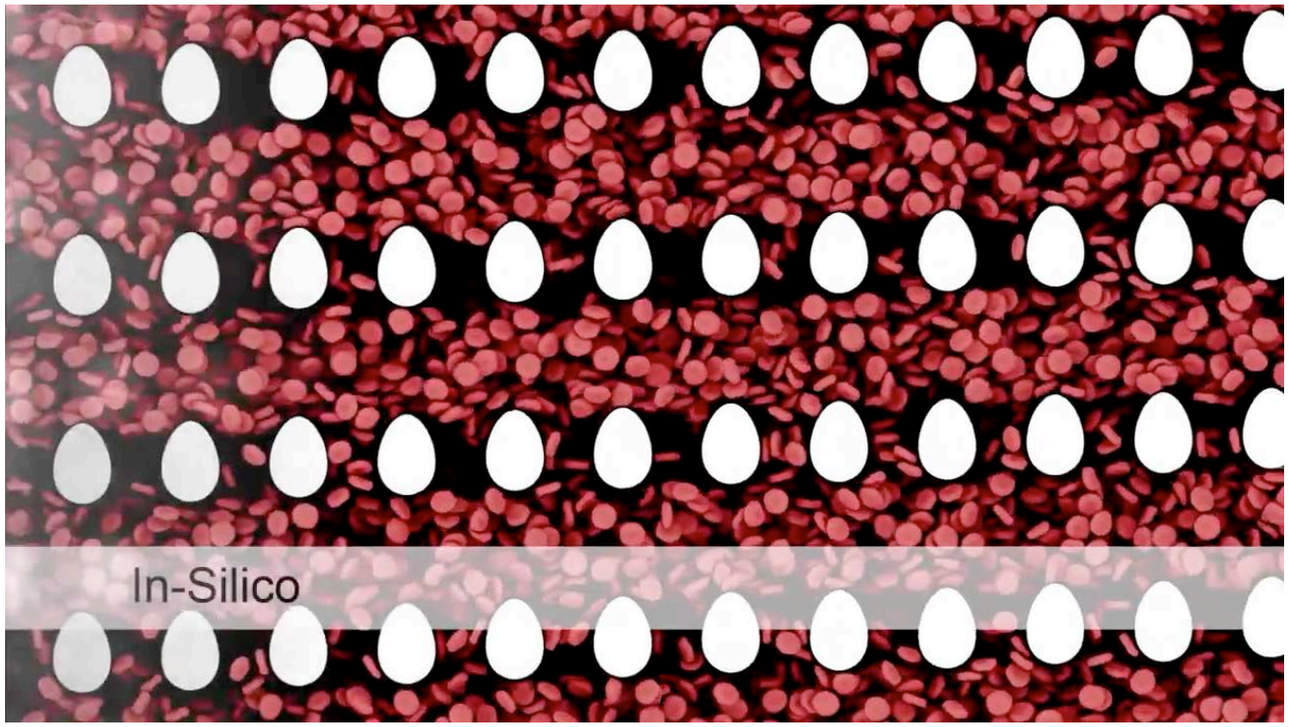
Exploiting the power of computation to resolve major challenges at the frontiers of engineering, natural and life sciences



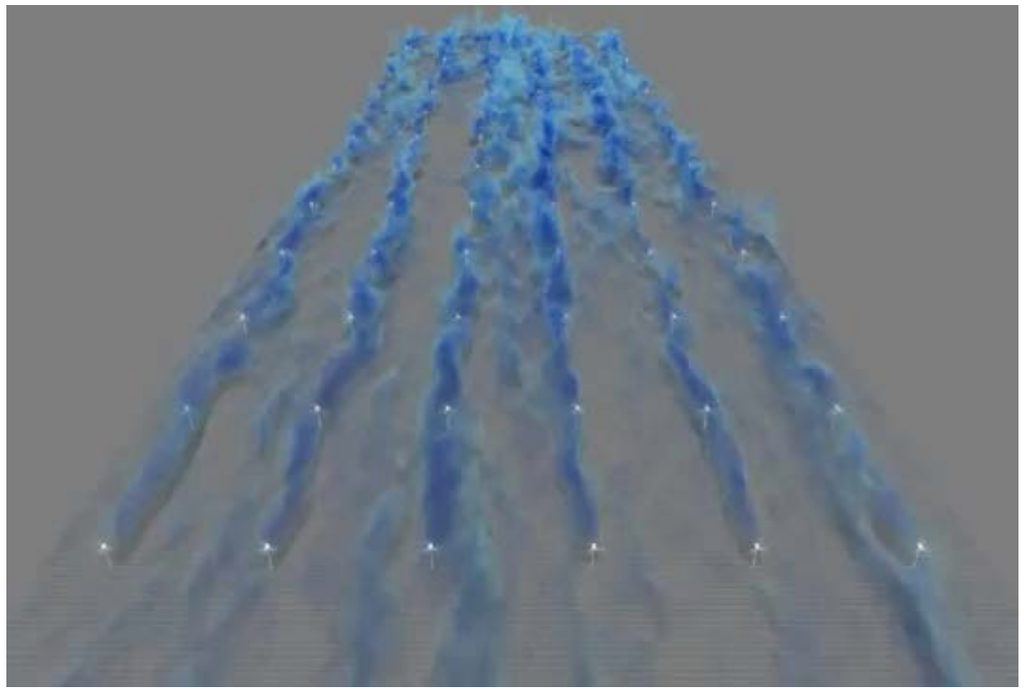
Simulation of Orion Spacecraft Launch Abort System (NASA Ames)



Detailed flow around an aircraft's landing gear (NASA Ames)



In-Silico Lab-on-a-Chip: high-throughput simulations of micro-fluidics at cell resolution (ETH Zurich)



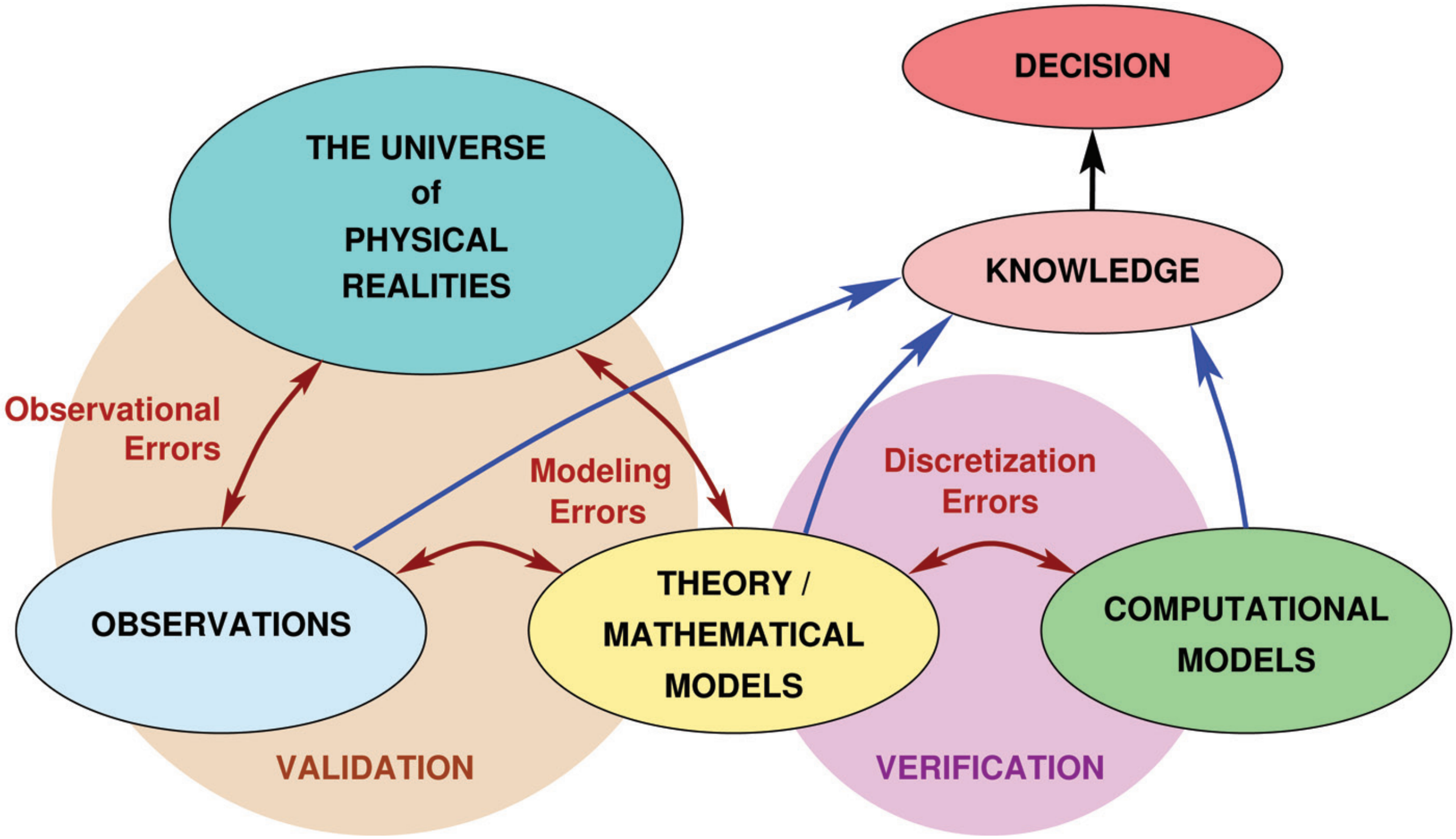
Flow field in a simulated wind-farm (JHU)

All models are wrong

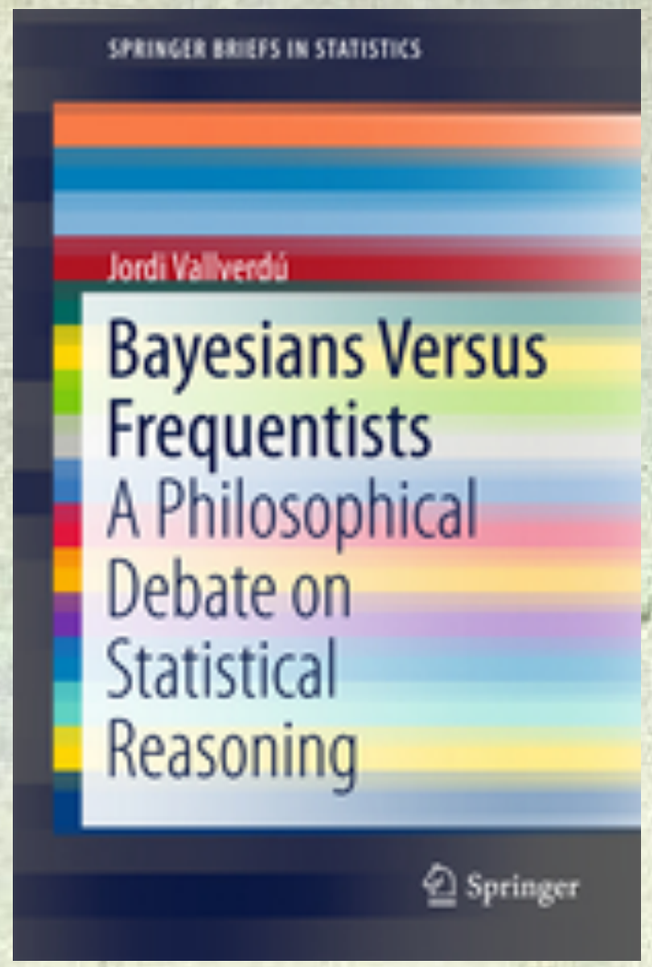
but some are useful

George E.P. Box

Predictive science



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



Aleatoric vs Epistemic Uncertainty

Course goals

- Learning how to analyze and synthesize data towards enhancing their understanding and ability to model physical, biological, and engineering systems.
- Hands-on skills on contemporary machine learning tools enabling them to construct prediction models, extract patterns and characterize the statistical properties of data.
- Applications of these tools spanning a diverse set of engineering disciplines, including fluid dynamics, heat transfer, mechanical design, and biomedical engineering.

<u>Key motifs:</u>

- Representation/Function approximation
- Optimization
- Uncertainty quantification

Disclaimers & FAQs

Is this the right course for me?

Yes, if you:

(a) Fulfill the pre-requisites (linear algebra, probability & statistics, optimization, Python programming).
(b) Want to build foundational knowledge in ML for engineering research and applications.
(c) Want to sharpen your implementation skills.

No, if you:

(a) Do not fulfill the pre-requisites.
(b) Are solely interested in theory.
(c) Are solely interested in CS applications (e.g. computer vision, NLP).
(d) Are solely interested in specialized topics (e.g. RL, geometric deep learning).

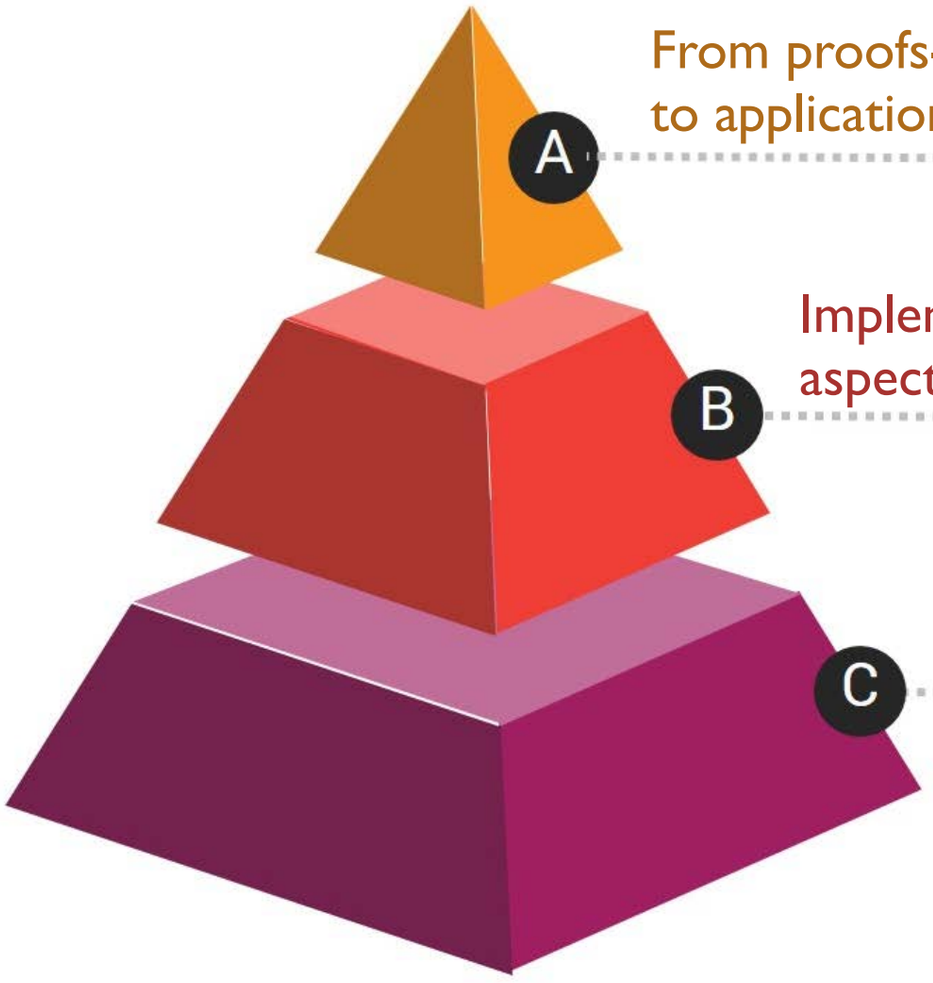
I am an experimentalist, what should I expect to learn?

(a) How to build predictive models from observational data.
(b) How to judiciously design and automate experiments.
(c) How to distill governing laws from experimental data.
(d) How to deal with aleatoric uncertainty

I am a computational scientist, what should I expect to learn?

(a) How to accelerate the prediction of large-scale and costly simulations.
(b) How to design and automate experiments.
(c) How to calibrate large-scale and costly computational models.
(b) How to deal with epistemic uncertainty.

Course philosophy



From proofs-of-concept to applications

80% of the homework assignments will be computational primarily based on proof-of-concept studies. Final projects will be geared towards applications.

Implementation aspects

20% of our lecture time will be devoted on hands-on programming tutorials.

Mathematical foundations

80% of our lecture time will be devoted on discussing the mathematical foundations of modern ML methods.

Assignments Grade Overall

Your grade is based on your performance during the semester. It includes readings, participation and hands-on tutorials in class (that means you must attend all classes), as well as a final project assignments, broken down as follows.

Assignment	Percentage Value	
Homework	40%	computational proof-of-concept studies
Midterm exam	20%	mathematical foundations
Final project	40%	applications
TOTAL	100%	

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

Supervised learning

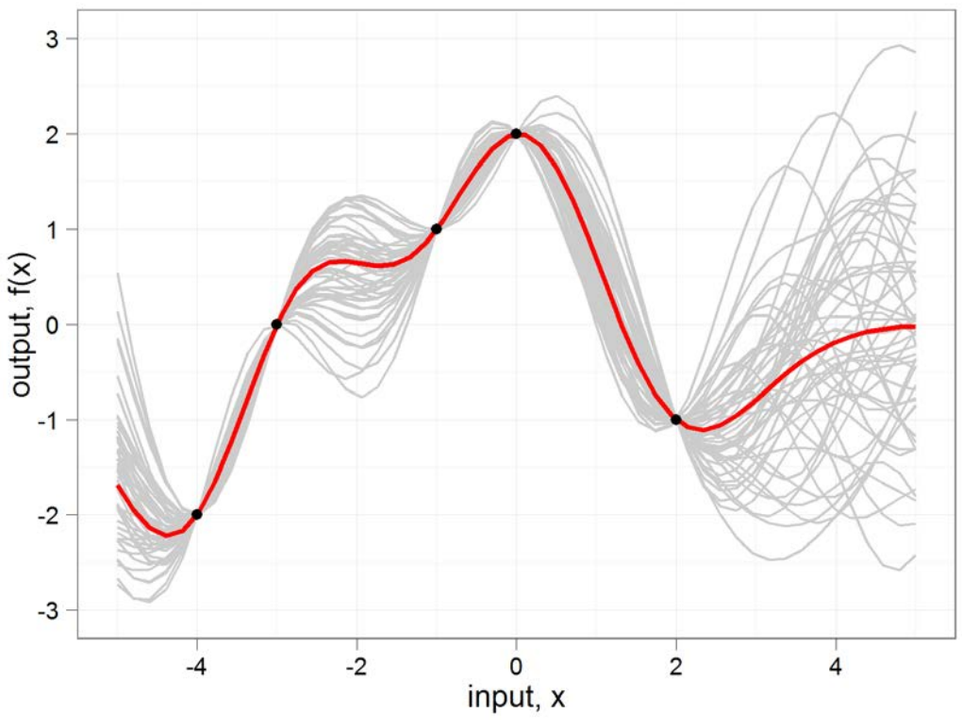
$$f : \mathcal{X} \to \mathcal{Y}$$

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

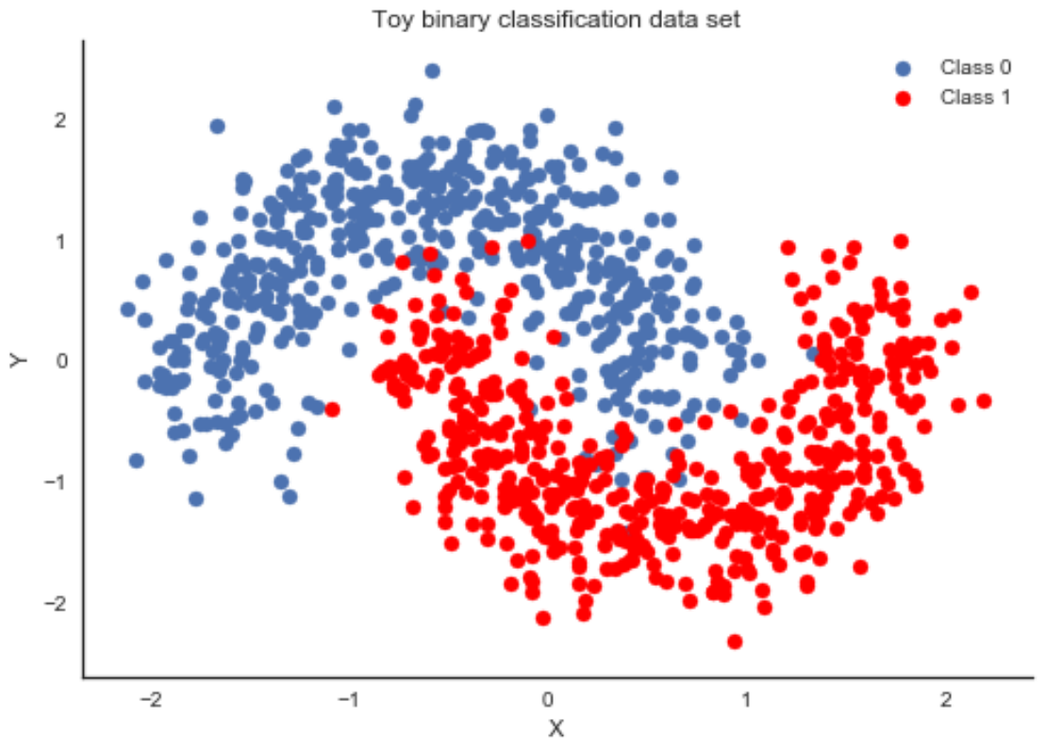
$$\boldsymbol{y} = f(\boldsymbol{x}) + \epsilon$$

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

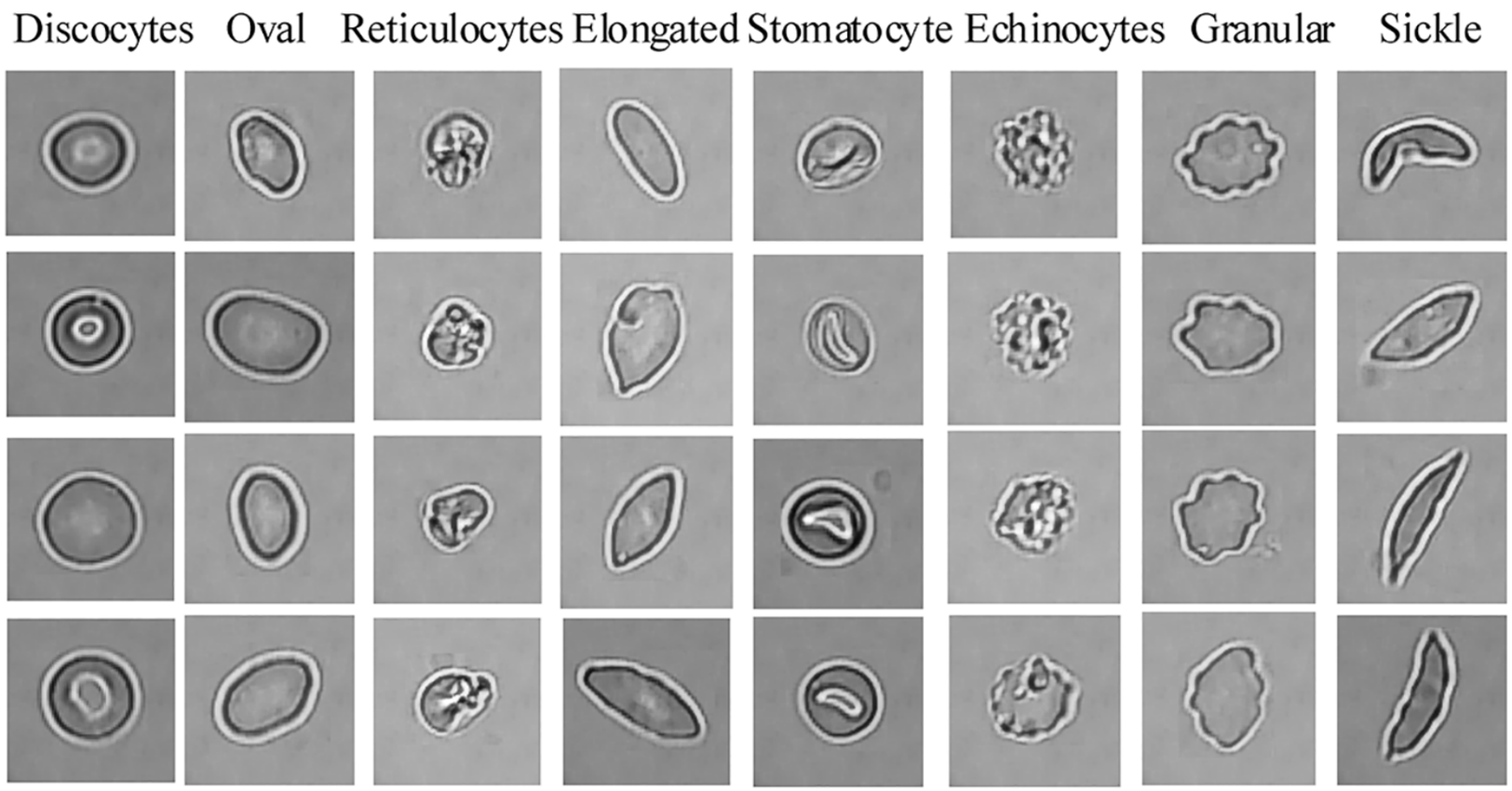
Regression



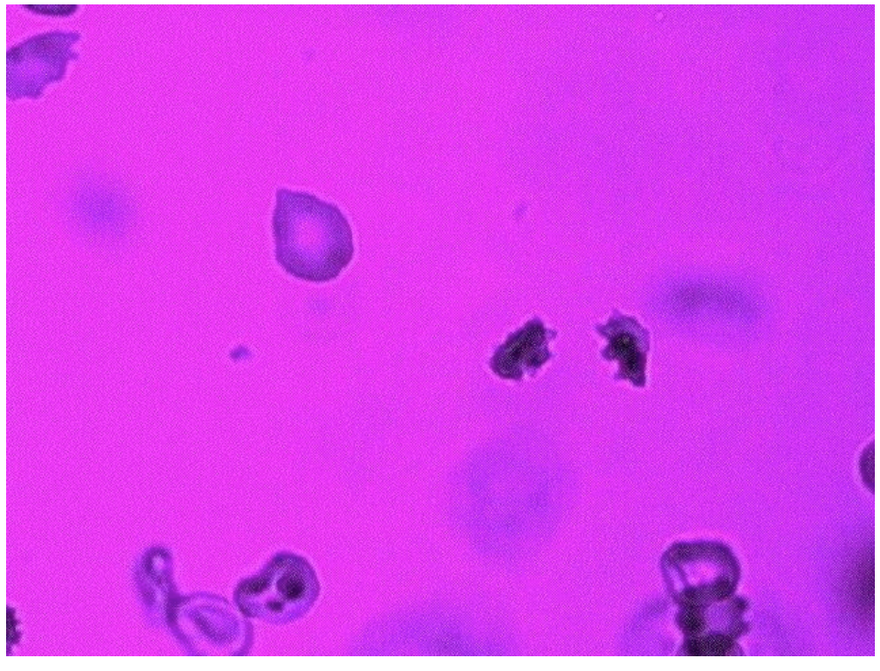
Classification



Classification



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



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

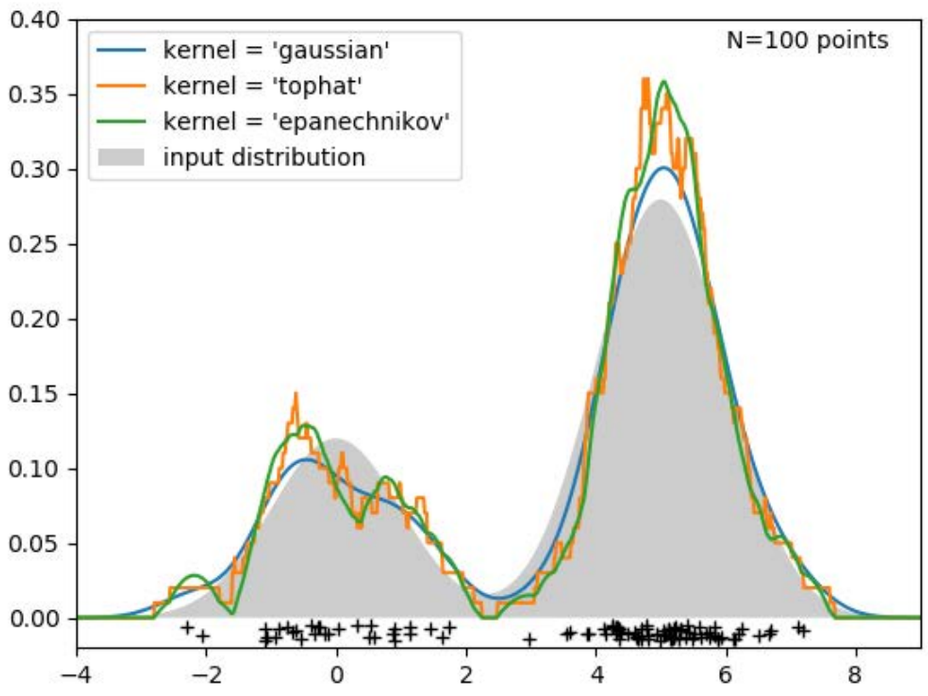
Unsupervised learning

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

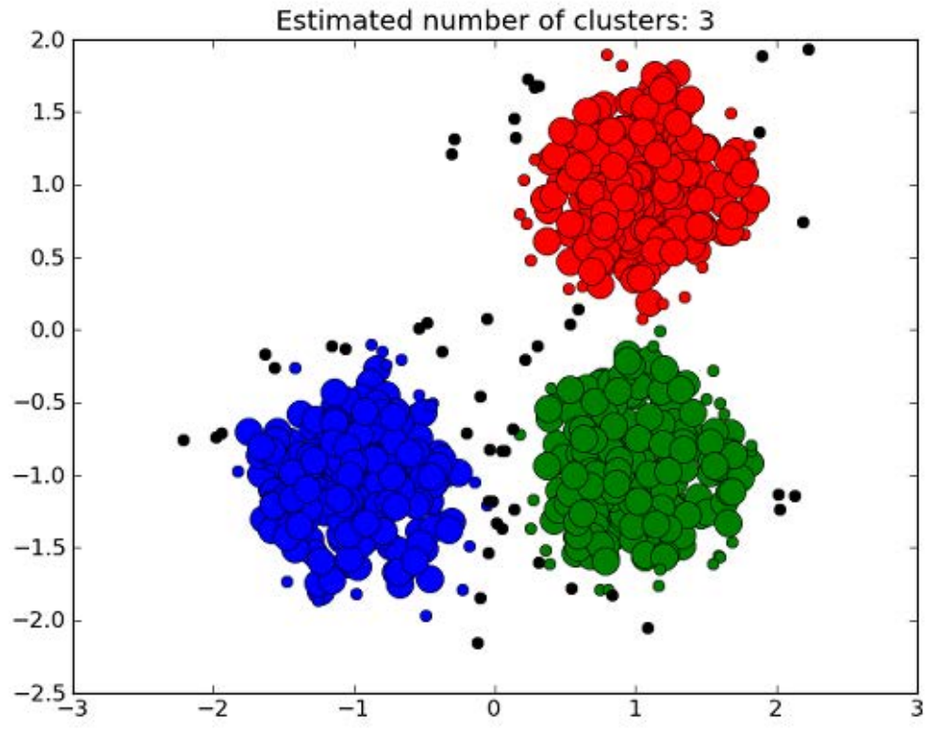
$$\boldsymbol{y} = f(\boldsymbol{z}) + \epsilon$$

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

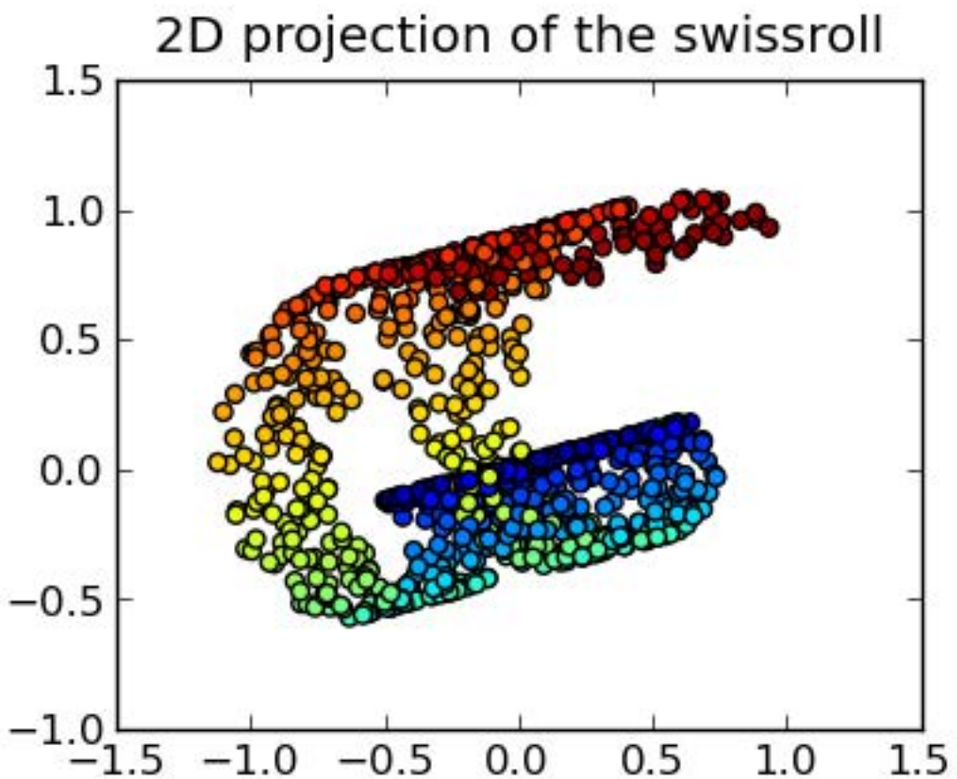
Density estimation

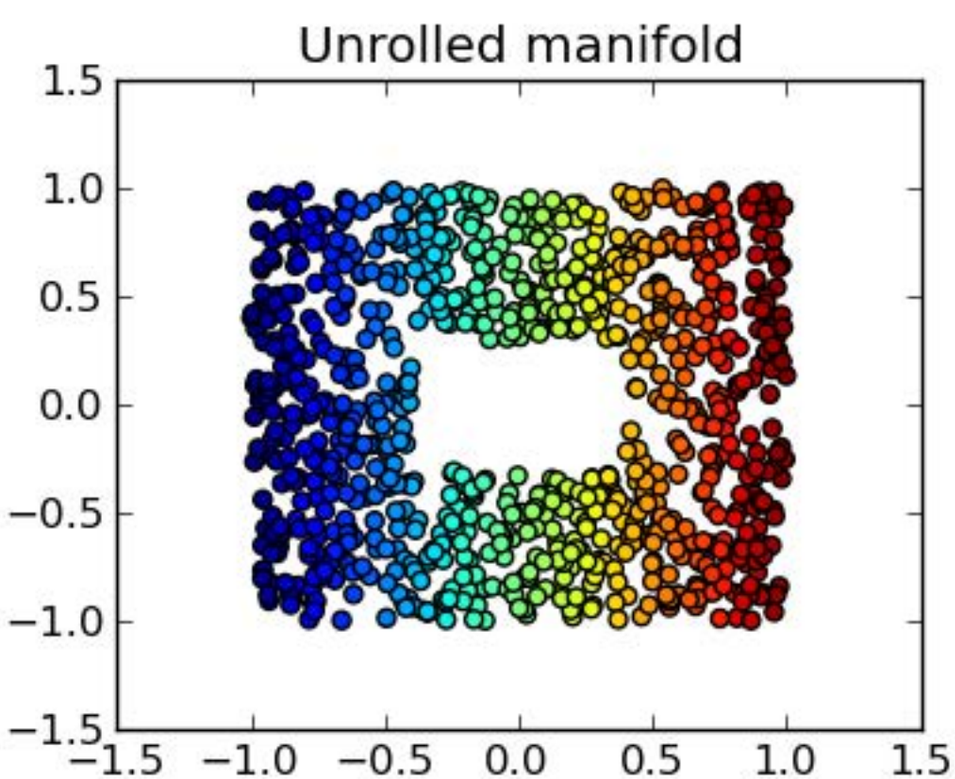


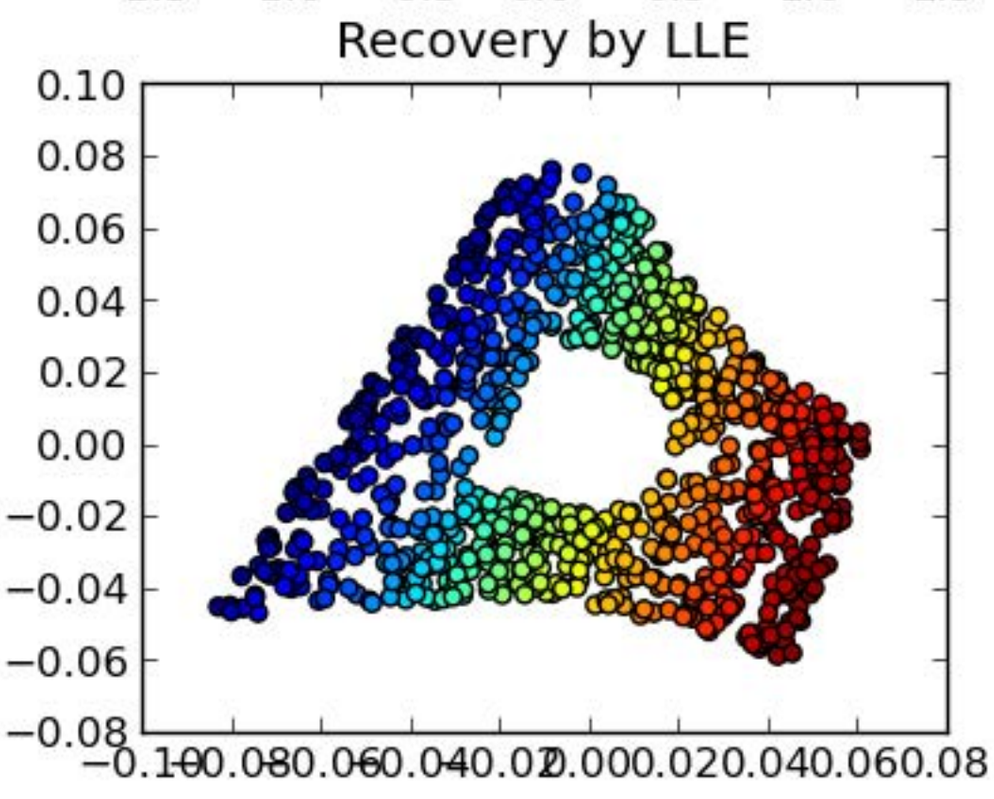
Clustering

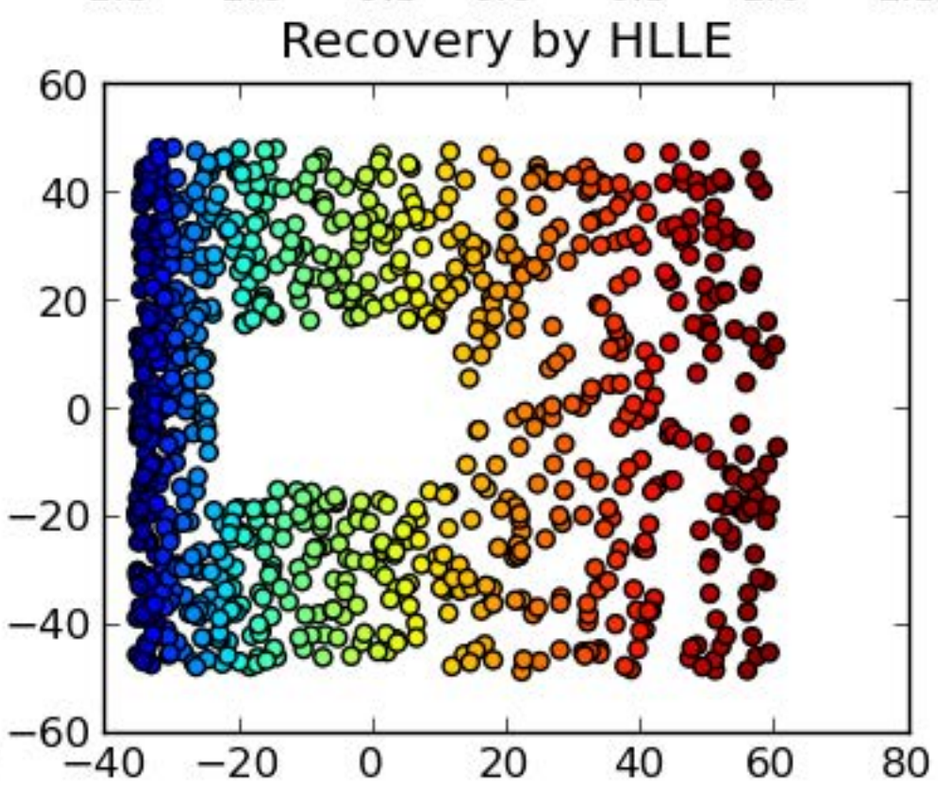


Dimensionality reduction



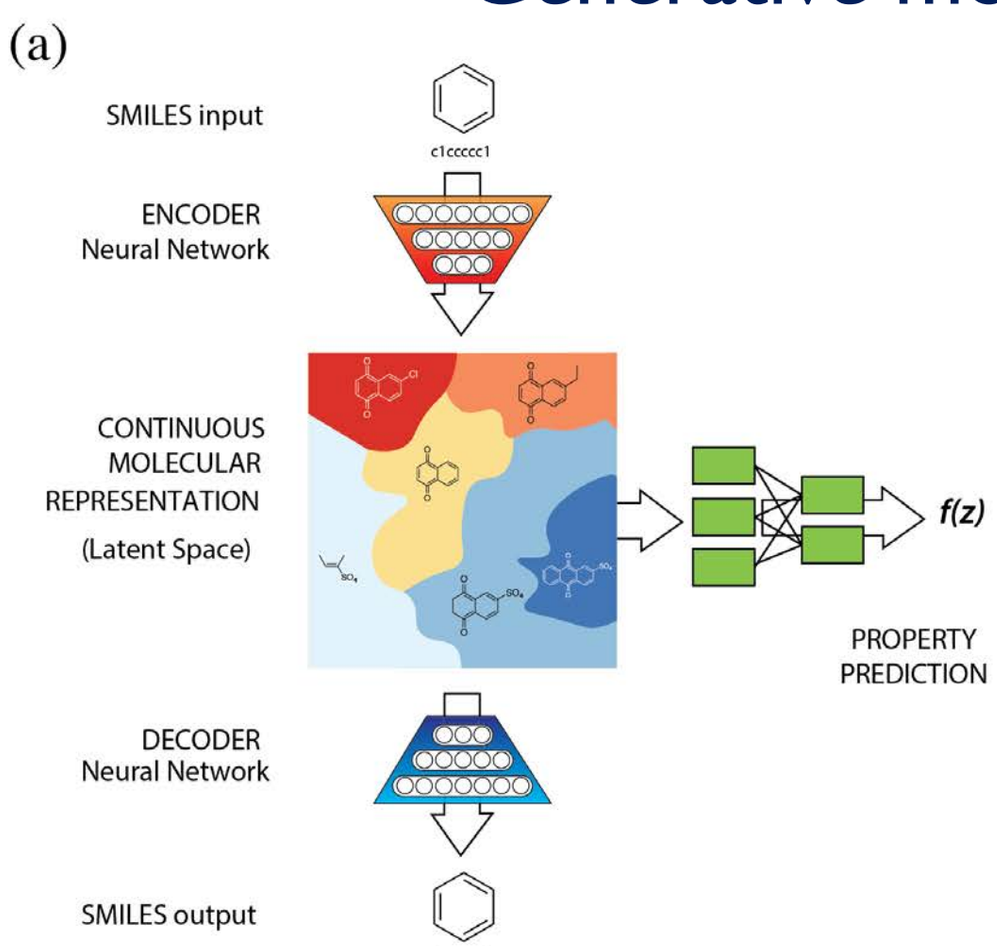


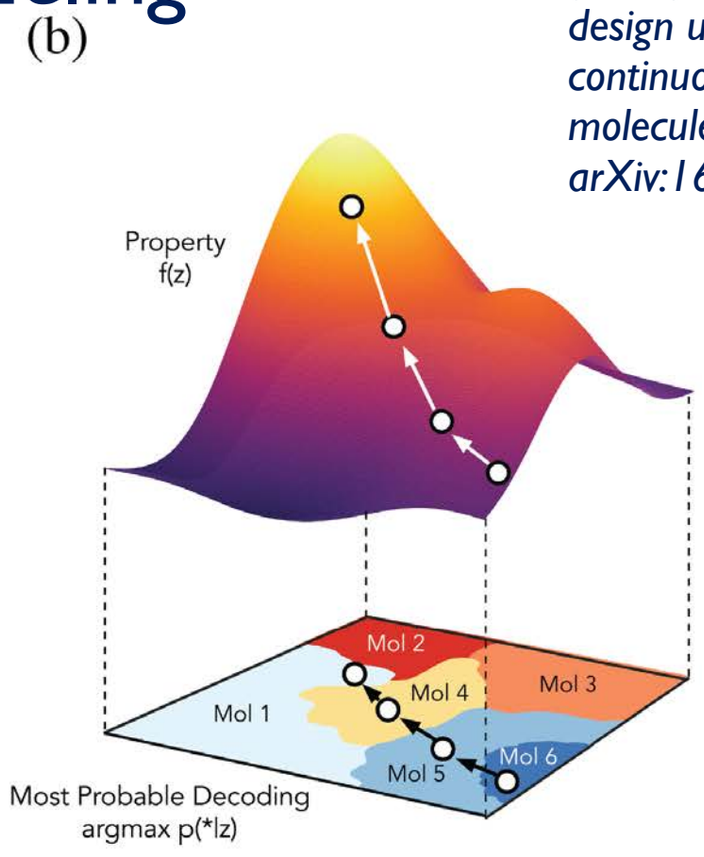




Generative modeling

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





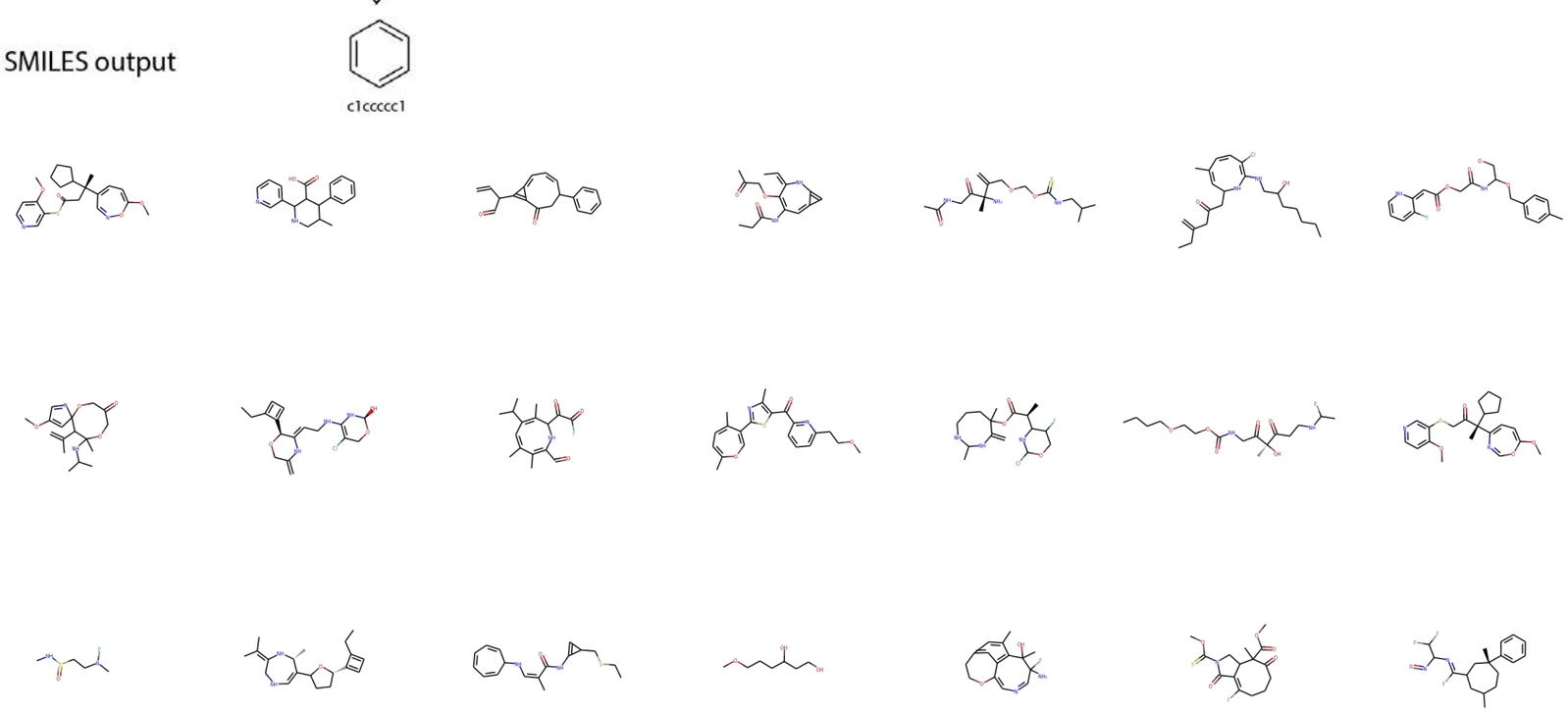


Figure 8: Molecules decoded from randomly-sampled points in the latent space of the ZINC VAE.

Course logistics

- Duration: 14 Weeks
- Time: Tuesdays and Thursdays, 3:30pm to 5:00pm (Towne 309)
- Dates: January 12 to April 26, 2022
- Instructor office hours: Thursdays 11:30 am to 1:30 pm
- TA office hours: 3-4pm Wednesdays, 11am-12pm Fridays
- Weekly/bi-weekly HW assignments, mid-term exam, final project.

For more details visit the course website:
https://www.seas.upenn.edu/~enm5310/

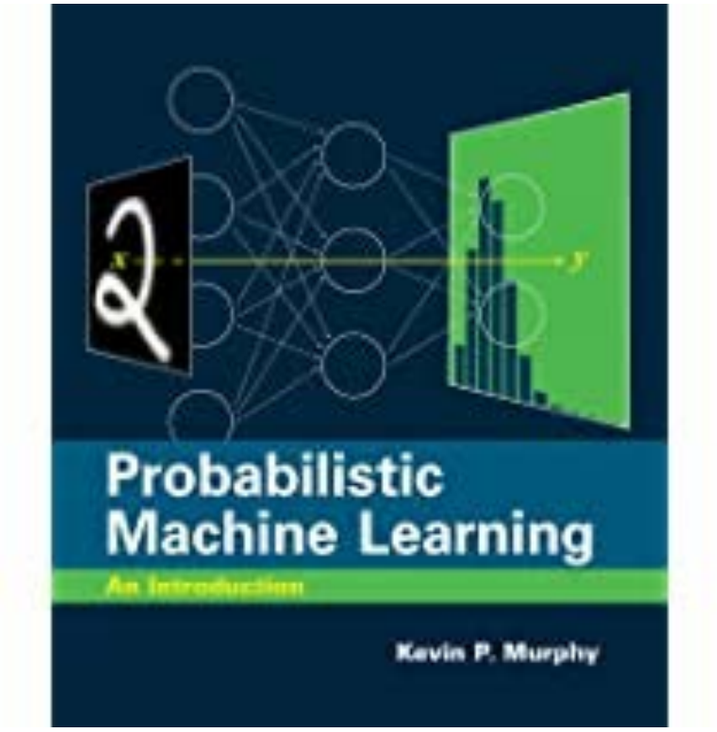
Slides and code can be found on Github:
https://github.com/PredictiveIntelligenceLab/ENM5310

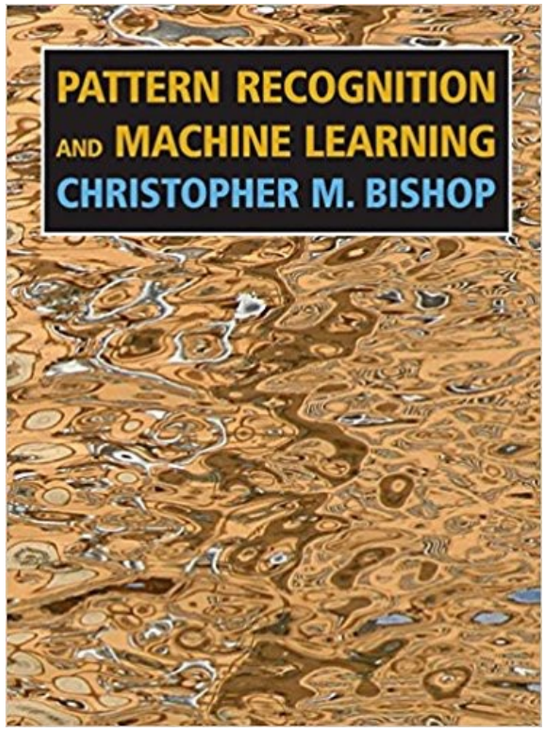
(Tentative) Schedule

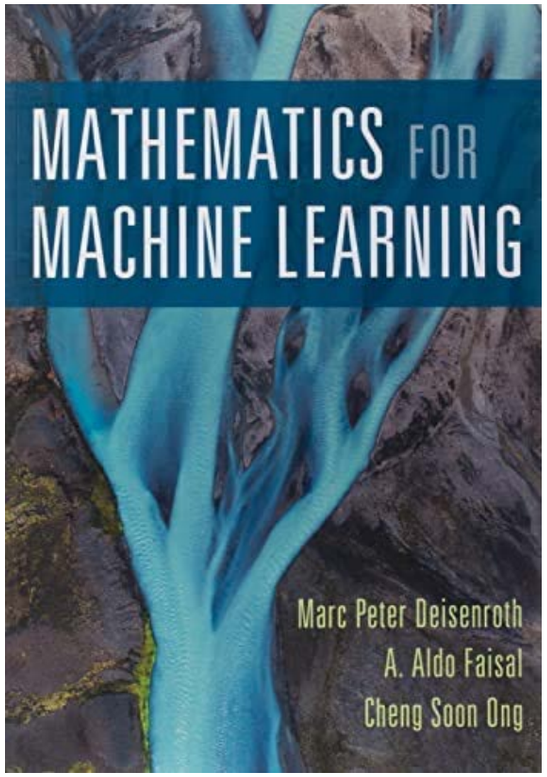
- Week 1: Probability and statistics recap
- Week 2: Statistical inference: Maximum likelihood estimation, maximum a-posteriori estimation
- Week 3: Optimization recap: gradient descent, Newton's algorithm, stochastic gradient descent
- Week 4: Approximate inference: Variational methods
- Week 5: Approximate inference: Monte Carlo sampling
- Week 6: Multi-layer perceptrons.
- Week 7: Convolutional neural networks.
- Week 8: Recurrent neural networks.
- Week 9: Applications
- Week 10: Gaussian processes
- Week 11: Design of experiments, active learning, Bayesian optimization.
- Week 12: Principal component analysis
- Week 13: Latent variable models, variational auto-encoders
- Week 14: Generative adversarial networks.
- Week 15: Class presentations of final papers.

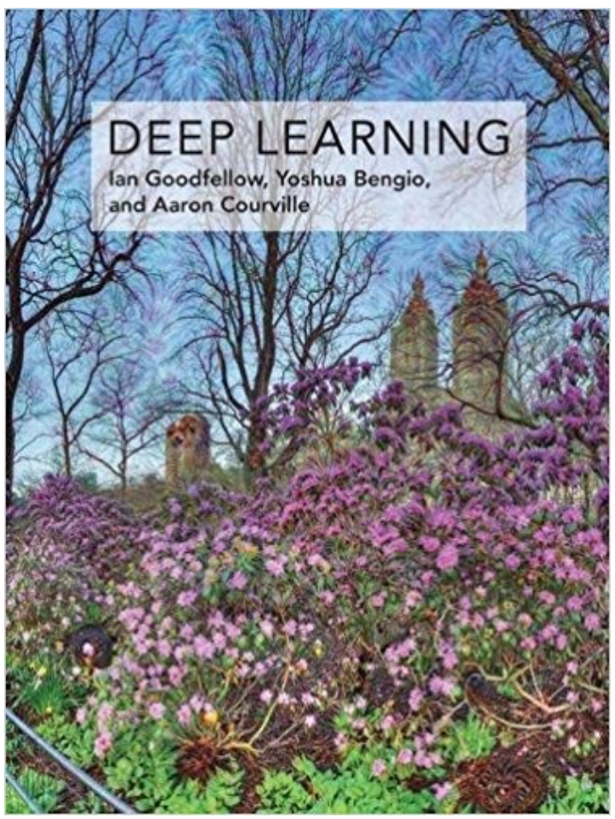
*not a strict breakdown

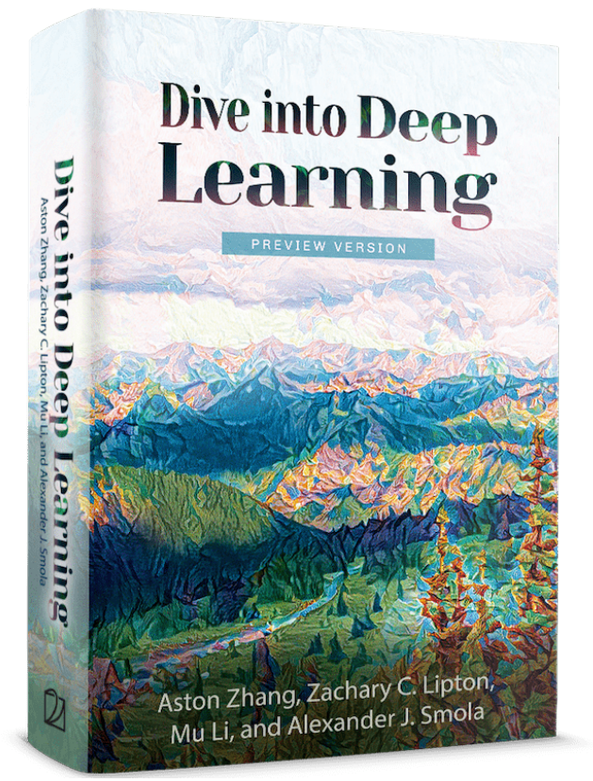
Textbooks

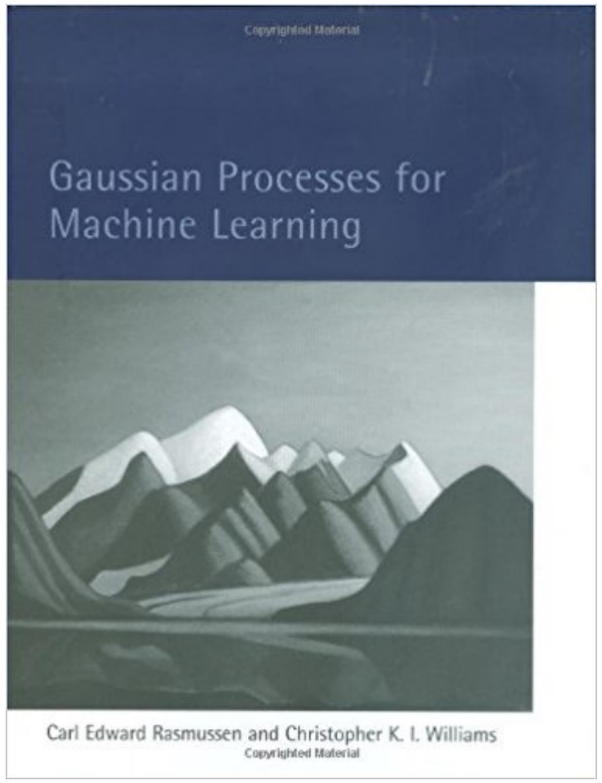

Probabilistic
Machine Learning
An Introduction
Kevin P. Murphy


PATTERN RECOGNITION
AND MACHINE LEARNING
CHRISTOPHER M. BISHOP


MATHEMATICS FOR
MACHINE LEARNING
Marc Peter Deisenroth
A. Aldo Faisal
Cheng Soon Ong


DEEP LEARNING
Ian Goodfellow, Yoshua Bengio,
and Aaron Courville


Dive into Deep
Learning
PREVIEW VERSION
Aston Zhang, Zachary C. Lipton,
Mu Li, and Alexander J. Smola


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

Reading

- Introduction
- Syllabus review
- Presentation of diverse applications that showcase the use of data, modeling, and scientific computation.
- Overview of the main themes and goals of this course
- Primer on Probability and Statistics

Reading

- The Emergence of Predictive Computational Science:
  - Computer predictions with quantified uncertainty (Oden, Moser, & Ghattas, 2010)
  - Lecture by J.T Oden.
- Review papers on recent advances in machine learning:
  - Probabilistic machine learning and artificial intelligence (Ghahramani, 2015)
  - Deep learning (LeCun, Bengio, & Hinton, 2015)
  - Machine learning: Trends, perspectives, and prospects (Jordan & Mitchell, 2015)
- Scientific computing in Python:
  - Lectures and code by Robert Johansson.

1. Oden, T., Moser, R., & Ghattas, O. (2010). Computer predictions with quantified uncertainty, part I. *SIAM News*, *43*(9), 1–3.
2. Ghahramani, Z. (2015). Probabilistic machine learning and artificial intelligence. *Nature*, *521*(7553), 452–459.
3. LeCun, Y., Bengio, Y., & Hinton, G. (2015). Deep learning. *Nature*, *521*(7553), 436–444.
4. Jordan, M. I., & Mitchell, T. M. (2015). Machine learning: Trends, perspectives, and prospects. *Science*, *349*(6245), 255–260.

Spyder

/Users/juanis/Documents/SpyderDocs/Flight_Operations.py

temp.py | Flight_Operations.py

```
#!/usr/bin/env python3
# -*- coding: utf-8 -*-
"""
Created on Mon May  4 17:07:07 2020

@author: juanis
"""

import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from scipy.spatial import ConvexHull, KDTree, Voronoi, voronoi_plot_2d
from shapely.geometry import Point, Polygon
plt.style.use('dark_background')


file_paths=["airports_CO.dat",
"borders_CO.dat"]

airports_CO=file_paths[0]
borders_CO=file_paths[1]


# Read files

airports_Col = pd.read_csv(airports_CO, sep=r"\s+", names=["coord-y", "coord-x",
                           'Altitude', "City", "Department", "Airport"])
borders_Col = pd.read_csv(borders_CO, sep=r"\s+", names=["coord-y", "coord-x"])


class FlightOperations:

    def __init__(self, airports, borders):
        self.airports = airports
        self.borders = borders
        self.points = self.airports[['coord-x', 'coord-y']].to_numpy()
        self.hull = ConvexHull(self.points)
        self.vor = Voronoi(self.points, )

    def plotAirports(self):
        """ Plot map with airports """
        voronoi_plot_2d(self.vor)
        plt.plot(self.borders['coord-x'], self.borders['coord-y'])
        plt.show()
```

/Users/juanis

Profile | Stop

Output | Save data | Load data | Clear comparison

Function/Modu	Total Time	Diff	Local Time	Diff	Calls

Variable explorer | Help | Plots | Files | Profiler

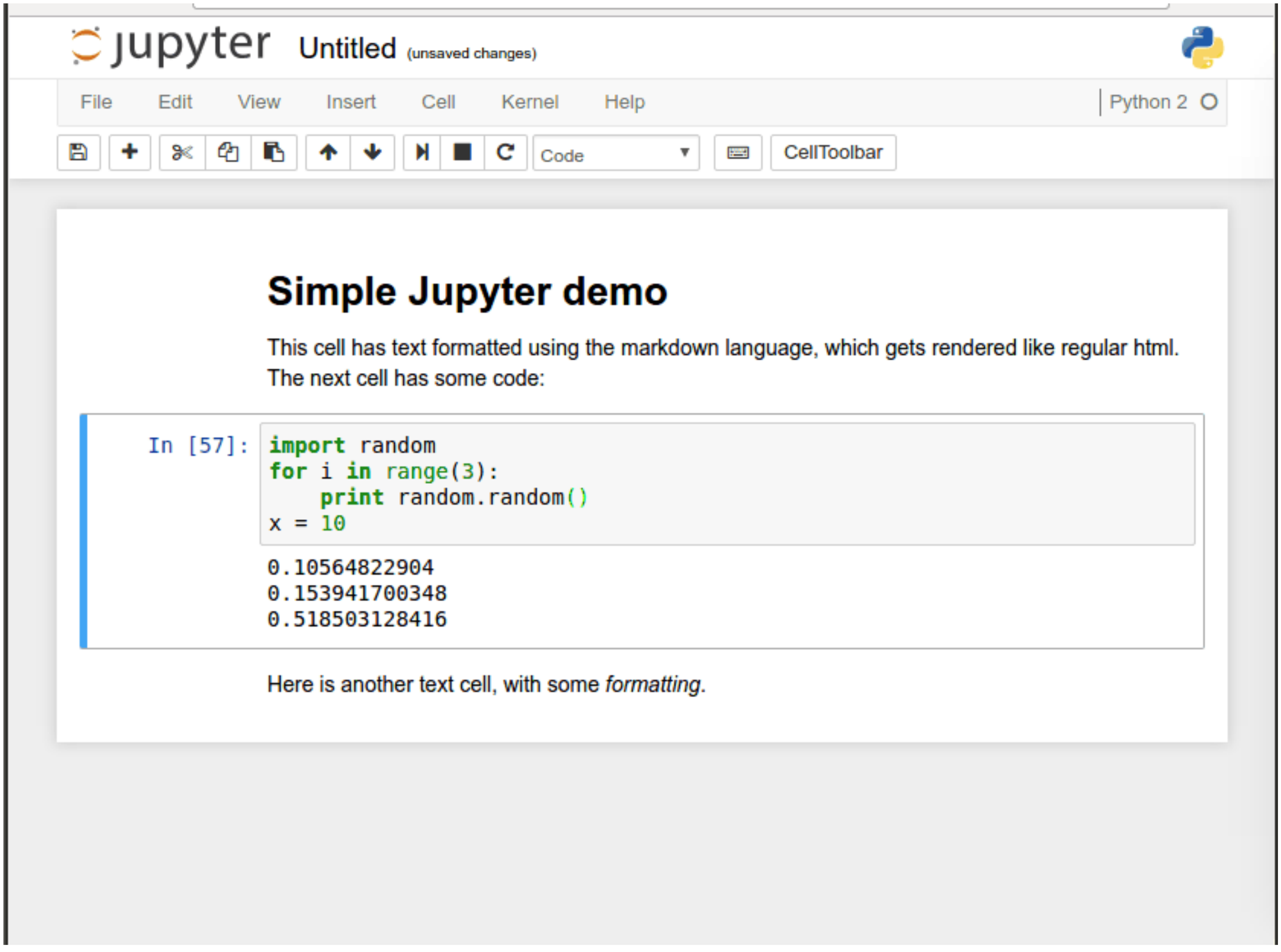
Console 1/A

```
Python 3.7.7 (default, Mar 26 2020, 10:32:53)
Type "copyright", "credits" or "license" for more information.

IPython 7.13.0 -- An enhanced Interactive Python.

In [1]:
```

IPython console | History

LSP Python: ready | conda: spyder-dev (Python 3.7.7) | Line 5, Col 1 | UTF-8 | LF | RW | Mem 55%

Jupyter notebook



Google colab

face_detection - Google Drive | face_detection.ipynb - Colaborat | colab.research.google.com/drive/1ss4n

Apps | Google | Varzesh3 | P30 | شرکت | PART Services | Paper/Code | trackers | Online courses | MCTS | RaspBerry Pi 4 | Gesture recognition | Cool Tricks | ROAD | Other bookmarks

face_detection.ipynb

File Edit View Insert Runtime Tools Help All changes saved

Comment | Share | m

Files

Upload | Refresh | Mount Drive

..

sample_data

+ Code + Text

RAM Disk | Editing

Using Google Colab to run our machine learning projects.

Change the directory to face_detection folder

First create a folder and name it as **face_detection**

```
from google.colab import drive
drive.mount('/content/drive/')
```

use `os.chdir` to connect to the face_detection directory

```
[ ] from os import chdir
    chdir('/content/drive/My Drive/face_detection')
```

Clone your project in the google drive

```
[ ] !git clone https://github.com/masouduut94/Pytorch_Retinaface.git
```

```
Cloning into 'Pytorch_Retinaface'...
remote: Enumerating objects: 26, done.
remote: Counting objects: 100% (26/26), done.
remote: Compressing objects: 100% (18/18), done.
remote: Total 146 (delta 7), reused 23 (delta 5), pack-reused 120
Receiving objects: 100% (146/146), 12.34 MiB | 24.73 MiB/s, done.
Resolving deltas: 100% (46/46), done.
```

Doing a little bit cleaning.

```
[ ] !mv Pytorch_Retinaface/* ./
    !rm -r Pytorch_Retinaface/
```

Get the pretrained weight files

1 - We open the link to the pretrained models.

2 - We right click on the weight file and choose "Add Shortcut to Drive"

Disk 37.15 GB available

https://colab.research.google.com/notebooks/intro.ipynb

Markov Chain Monte Carlo simulations in the 1950s

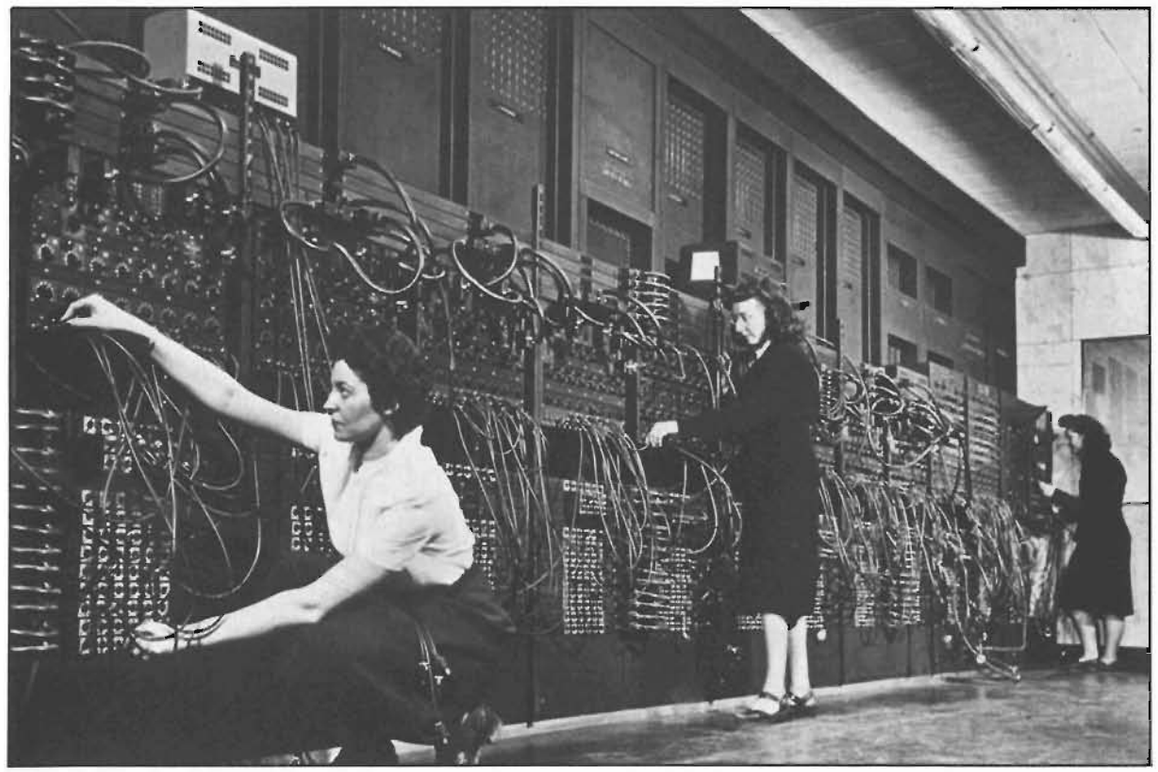


Fig. 1. Programming panels and cables of the ENIAC.

Pion-proton phase-shift analysis (Fermi, Metropolis; 1952)

Phase-shift analysis (Bethe, deHoffman, Metropolis; 1954)

Nonlinear coupled oscillators (Fermi, Pasta, Ulam; 1953)

Genetic code (Gamow, Metropolis; 1954)

Equation of state: Importance sampling (Metropolis, Teller; 1953)

Two-dimensional hydrodynamics (Metropolis, von Neumann; 1954)

Universalities of iterative functions (Metropolis, Stein, Stein; 1973)

Nuclear cascades using Monte Carlo (Metropolis, Turkevich; 1954)

Anti-clerical chess (Wells; 1956)

The lucky numbers (Metropolis, Ulam; 1956)

Fig. 6. Scientific triumphs achieved with the MANIAC. Nick Metropolis was a co-author of the publications resulting from these studies except for the ones on nonlinear coupled oscillators and anti-clerical chess.

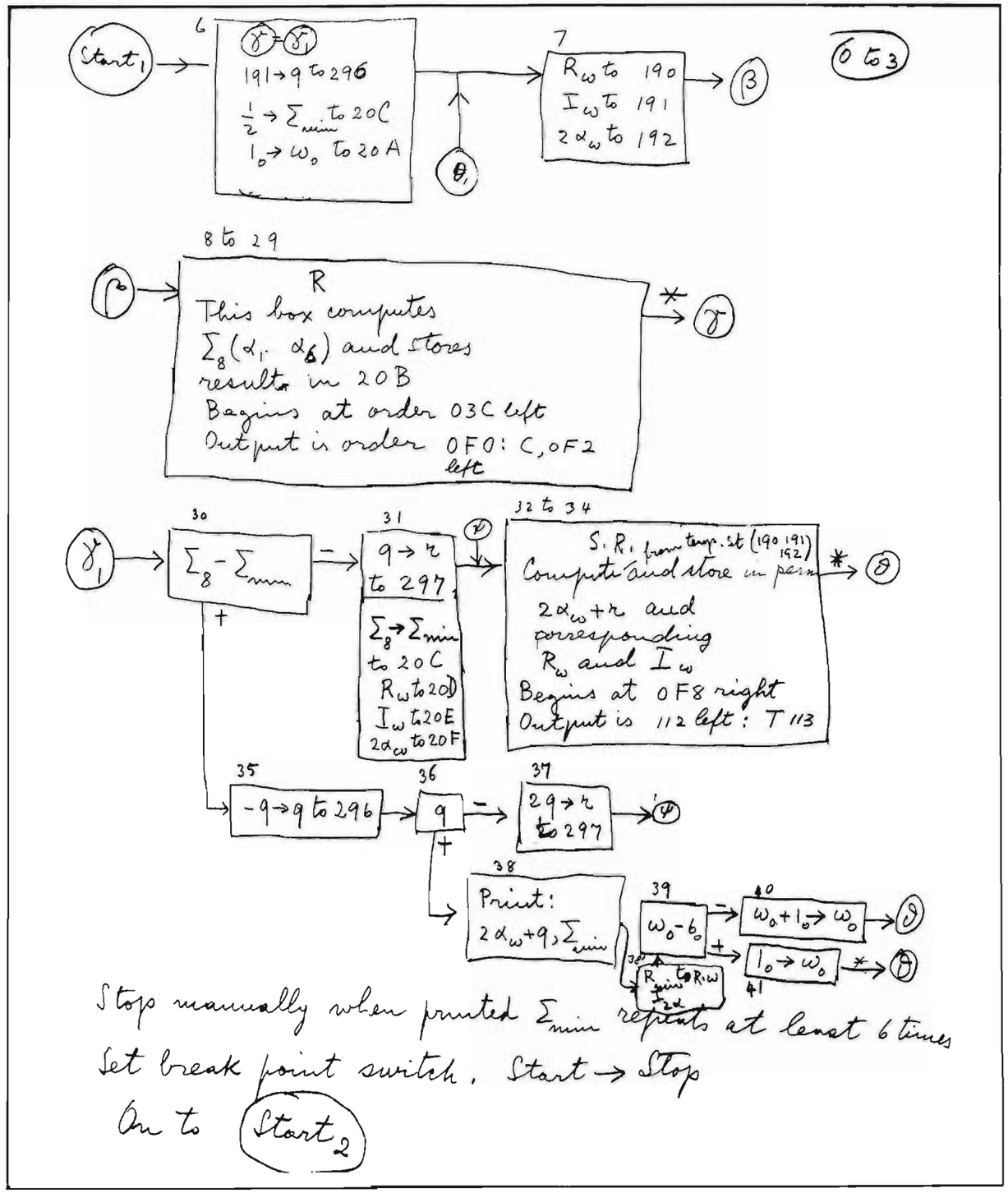


Fig. 4. A subprogram written by Fermi for calculating phase shifts by finding a minimum chi-squared in a fit to the data.