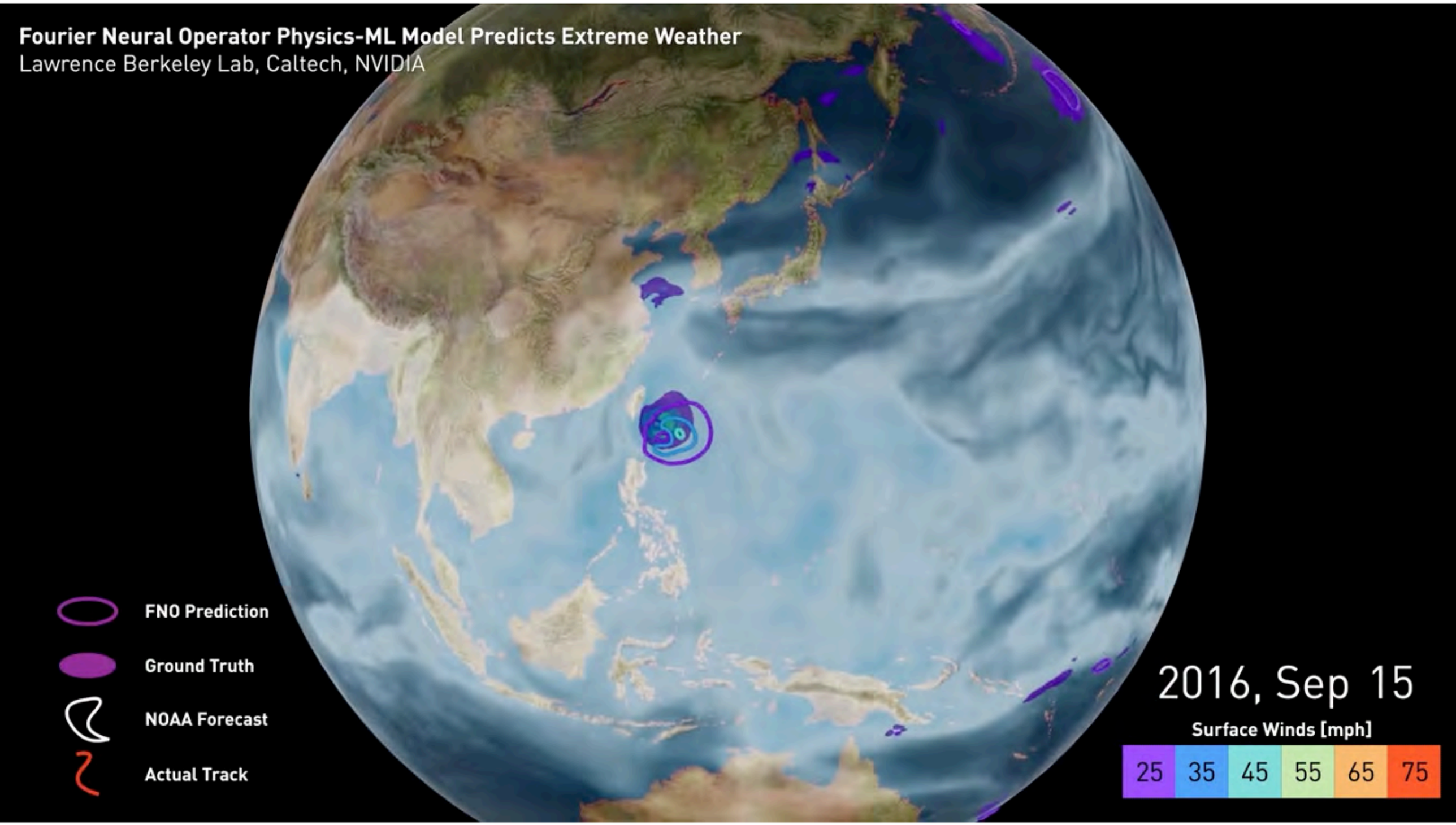
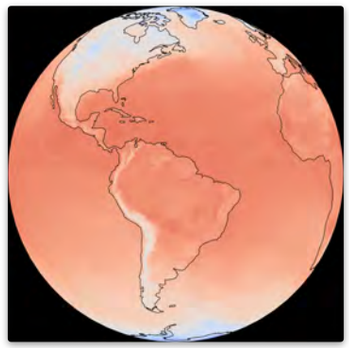
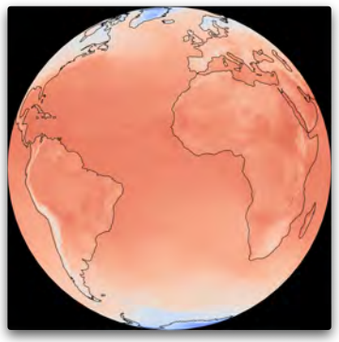
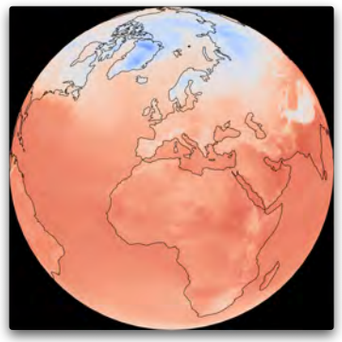
MEAM4600: AI for Science and Engineering

Lecture #0: Introduction, motivation and course logistics

Predictive science and engineering

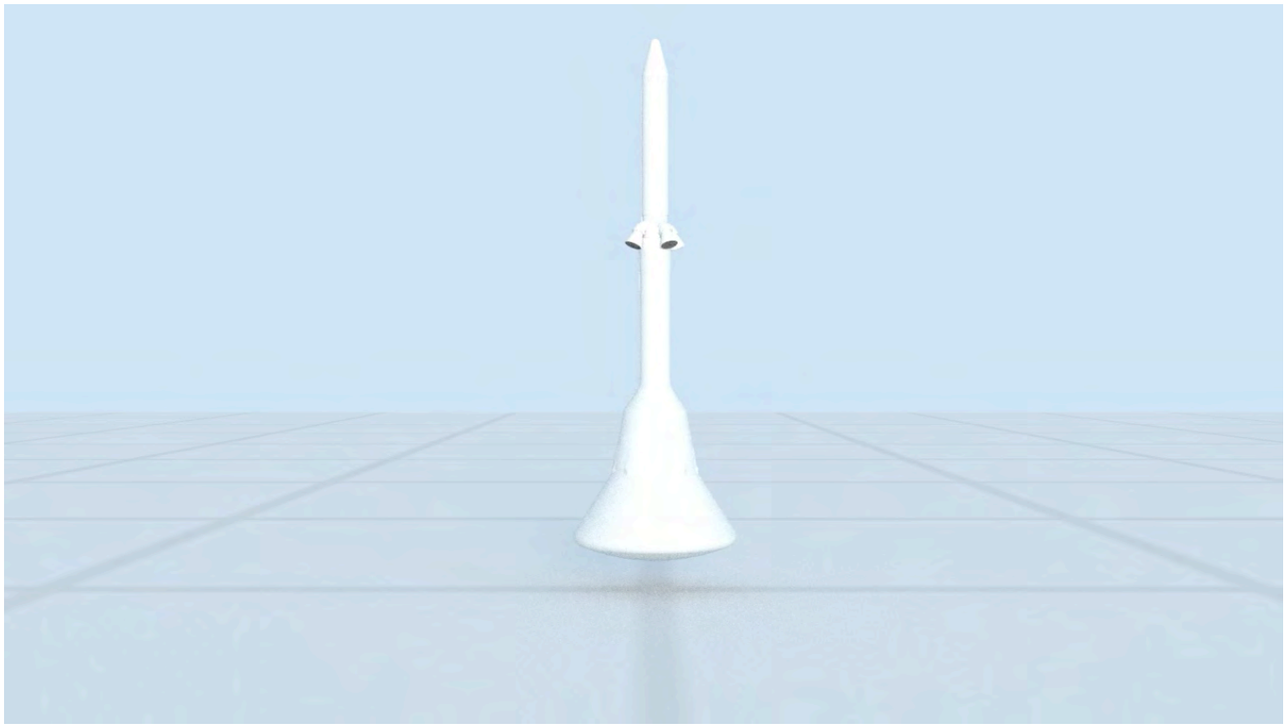


https://blogs.nvidia.com/blog/2021/11/16/ai-science-climate-change/

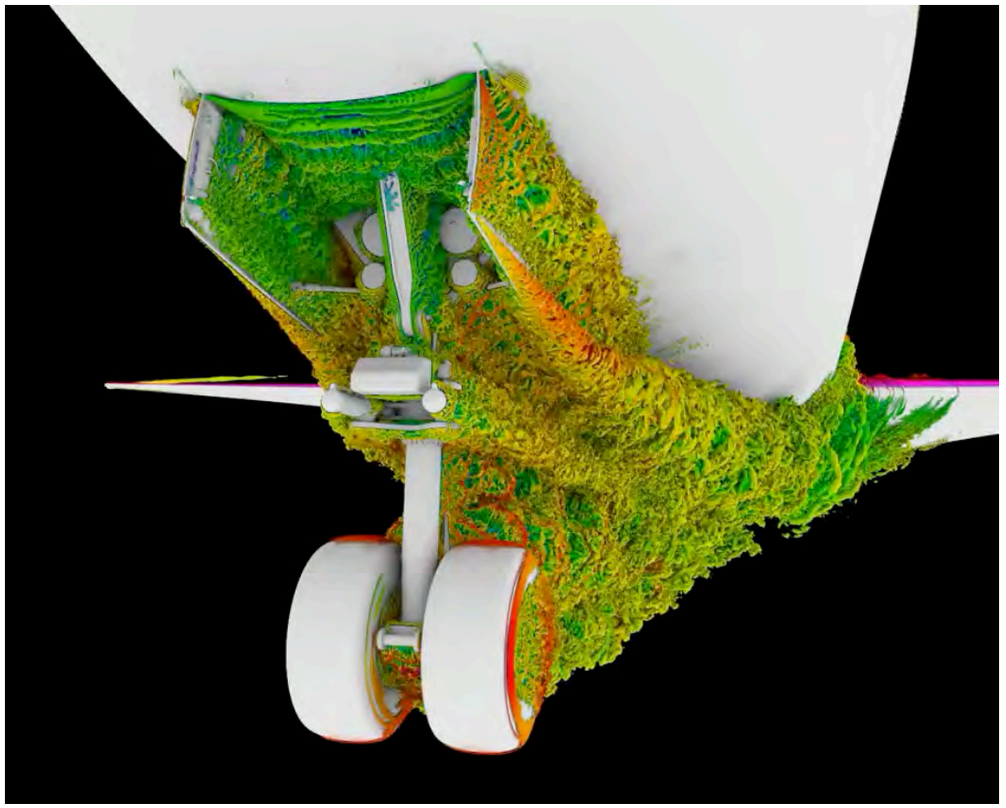




Computational science & engineering

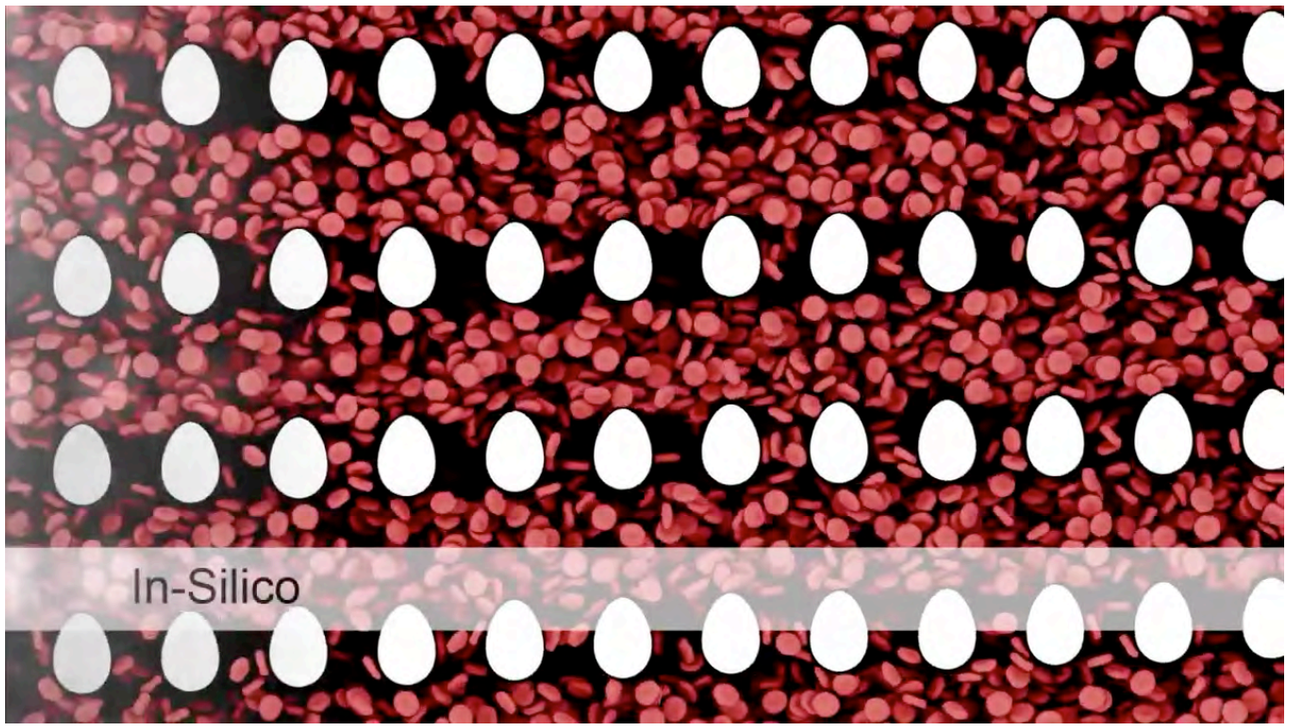
Exploiting the power of computation to resolve major challenges at the frontiers of engineering, natural and life sciences



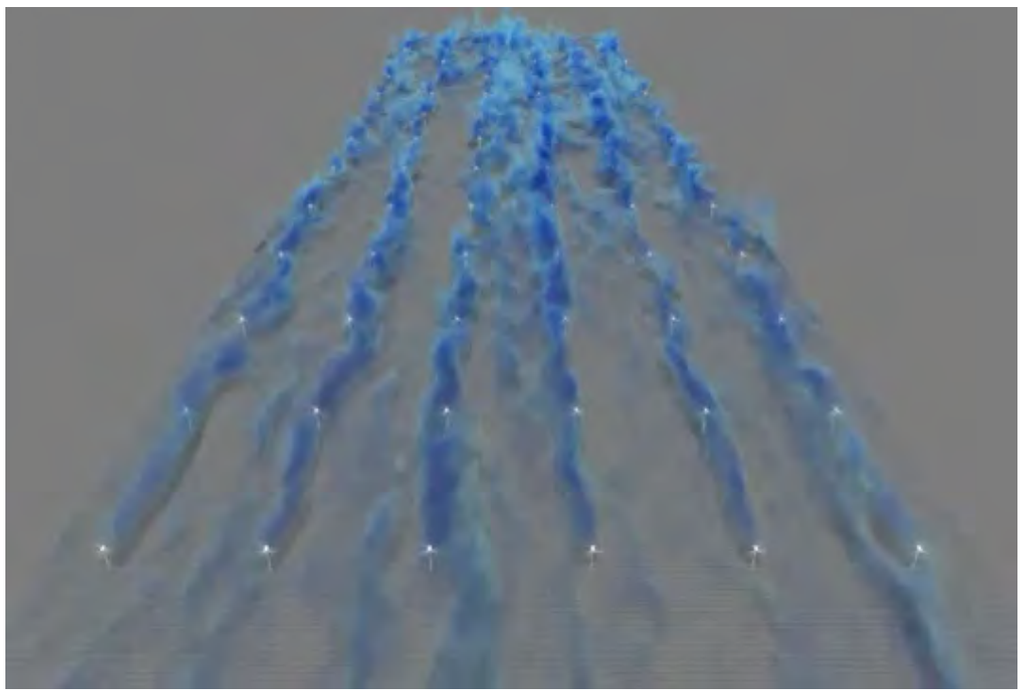
Simulation of Orion Spacecraft Launch Abort System (NASA Ames)



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



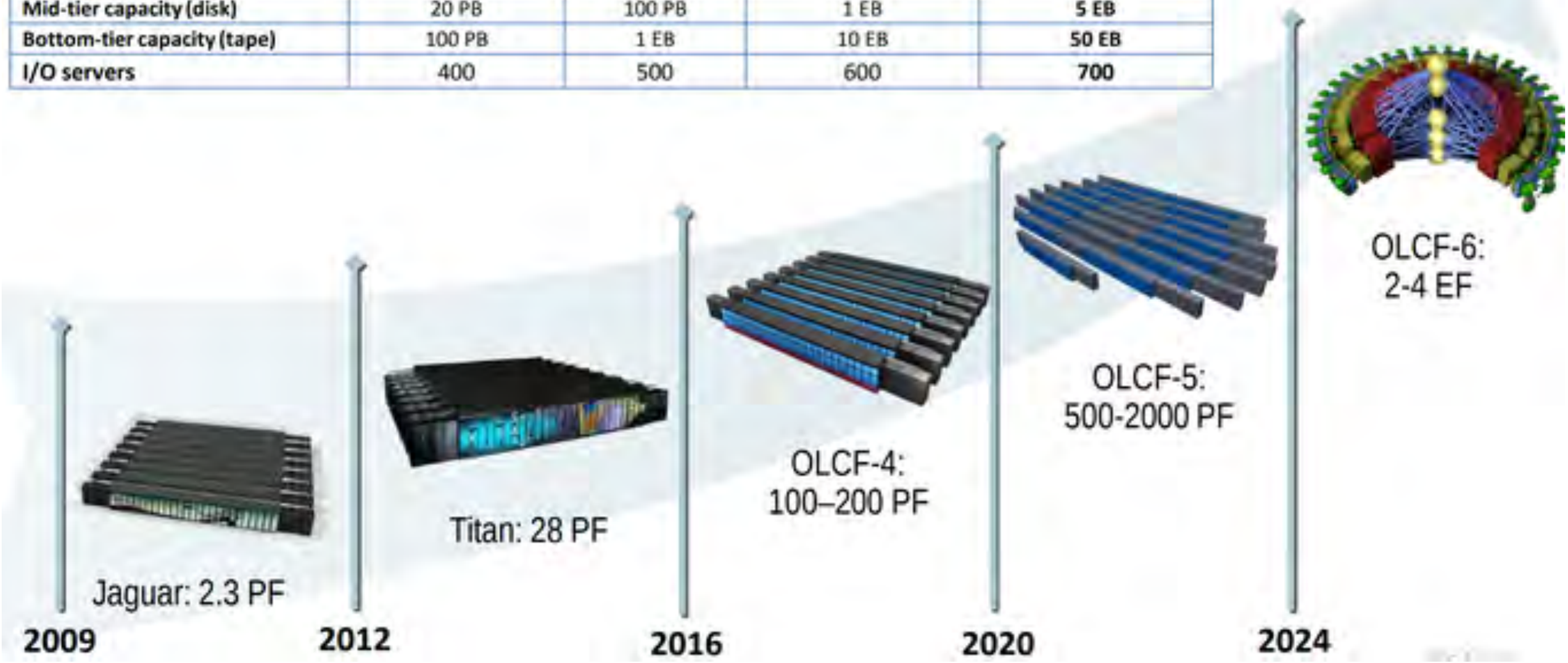
In-Silico Lab-on-a-Chip: high-throughput simulations of micro-fluidics at cell resolution (ETH Zurich)



Flow field in a simulated wind-farm (JHU)

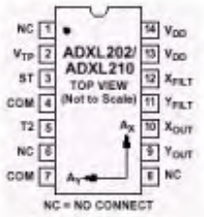
Roadmap to Exascale computing

	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.

Roadmap to Trillion Sensors



Accelerometer


Gyro


Pendulum Resistive
Tilt Sensors


Piezo Bend Sensor


Metal Detector

Gas Sensor


Gieger-Muller
Radiation Sensor


Digital Infrared Ranging

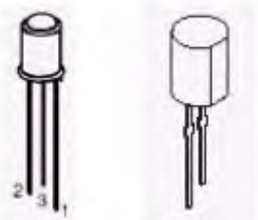

CDS Cell
Resistive Light Sensor

Resistive Bend Sensors

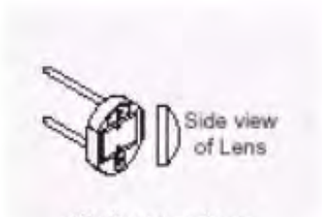

UV Detector


Pyroelectric Detector


Miniature Polaroid Sensor



IR Pin
Diode

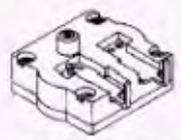


IR Sensor w/lens


Limit Switch


Mechanical Tilt Sensors


Touch Switch


Pressure Switch

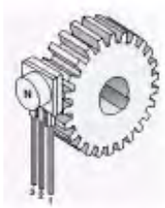

IR Reflection
Sensor


IR Amplifier Sensor


Thyristor

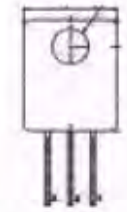

Magnetic Sensor

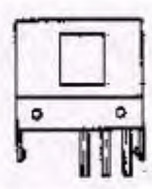

Magnetic Reed Switch


Hall Effect
Magnetic Field
Sensors


Polaroid Sensor Board


IRDA Transceiver


Lite-On IR
Remote Receiver


Radio Shack
Remote Receiver


IR Modulator
Receiver


Solar Cell

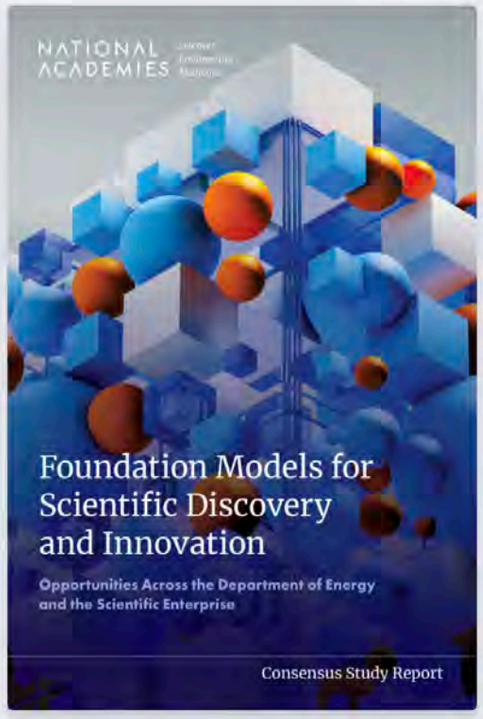

Compass


Compass


Piezo Ultrasonic Transducers

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

An Imminent Paradigm Shift



CONSENSUS

Foundation Models for Scientific Discovery and Innovation: Opportunities Across the Department of Energy and the Scientific Enterprise

2025

Read Online | Download PDF

Paperback – $21.60

Buy the book

Request an exam or desk copy

Foundation models - artificial intelligence systems trained on massive data sets to perform a wide range of tasks - have the potential to transform scientific discovery and innovation. At the request of the U.S. Department of Energy (DOE), the National Academies conducted a study to consider the capabilities of current foundation models as well as future possibilities and challenges. Foundation Models for Scientific Discovery and Innovation explores how foundation models can complement traditional computational methods to advance scientific discovery, highlights successful use cases, and recommends strategic approaches and investments to support DOE's mission.

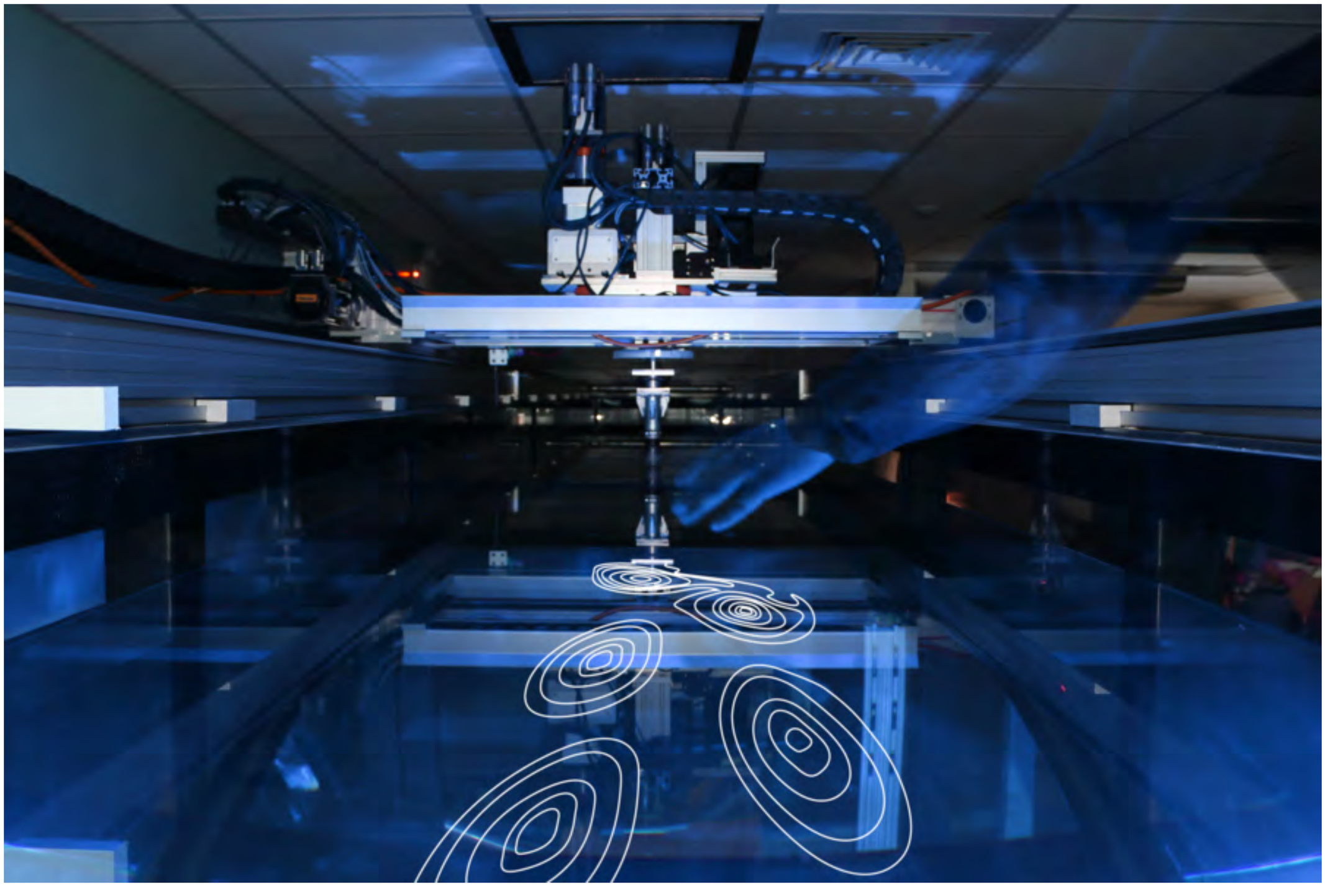
An Imminent Paradigm Shift

The US just launched a $100 Billion Manhattan Project for AI called Genesis

Autonomy

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

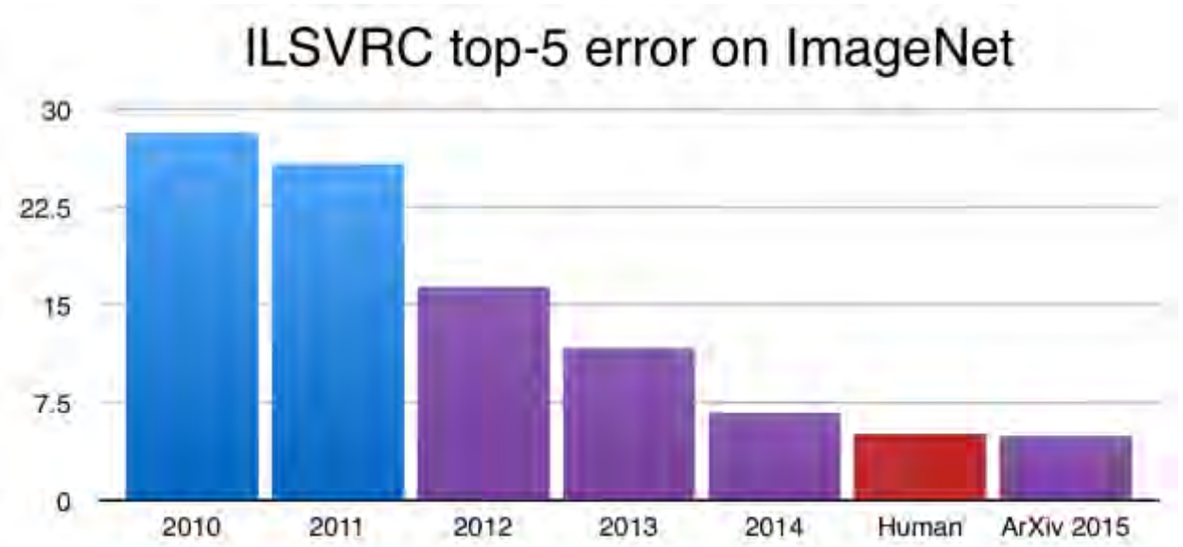
Intelligent experimentation

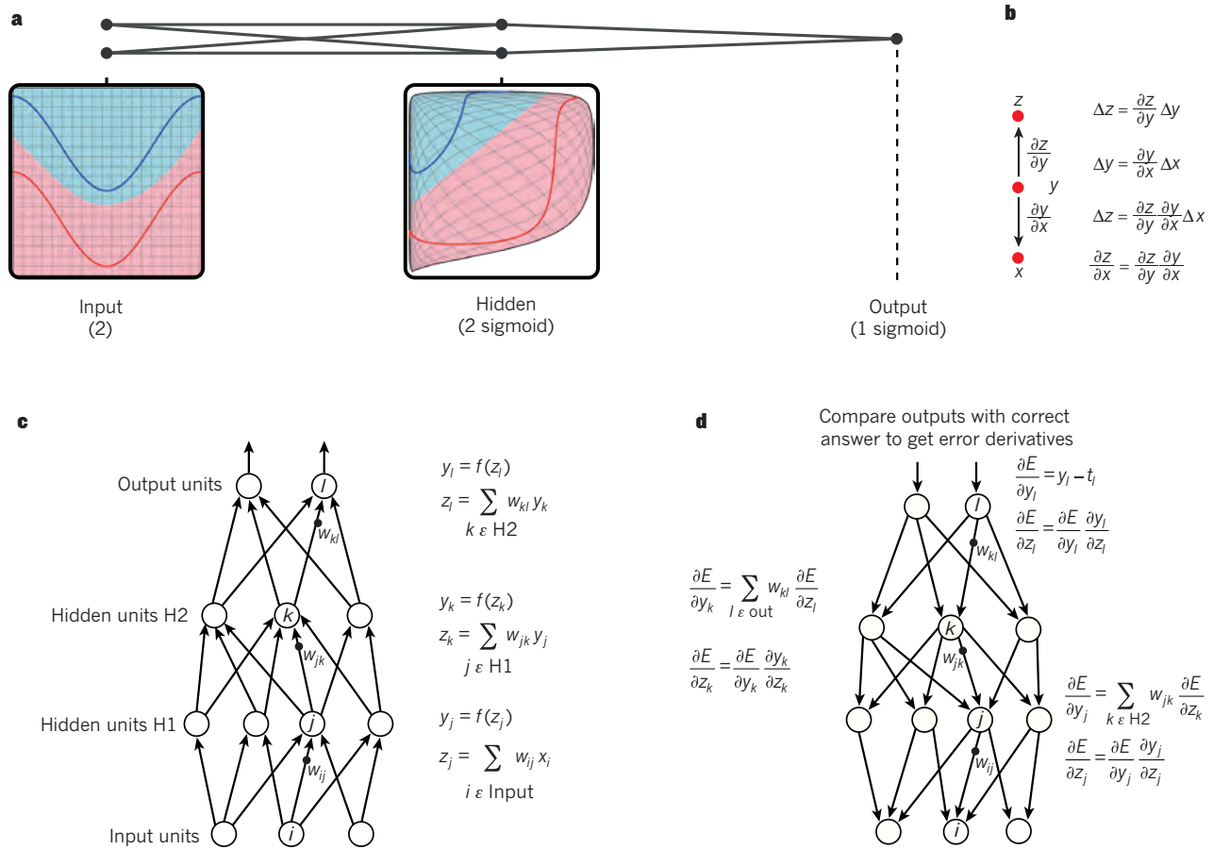


"In its first year of operation, the Intelligent Towing Tank (ITT) conducted about 100,000 total experiments, essentially completing the equivalent of a PhD student's five years' worth of experiments in a matter of weeks."

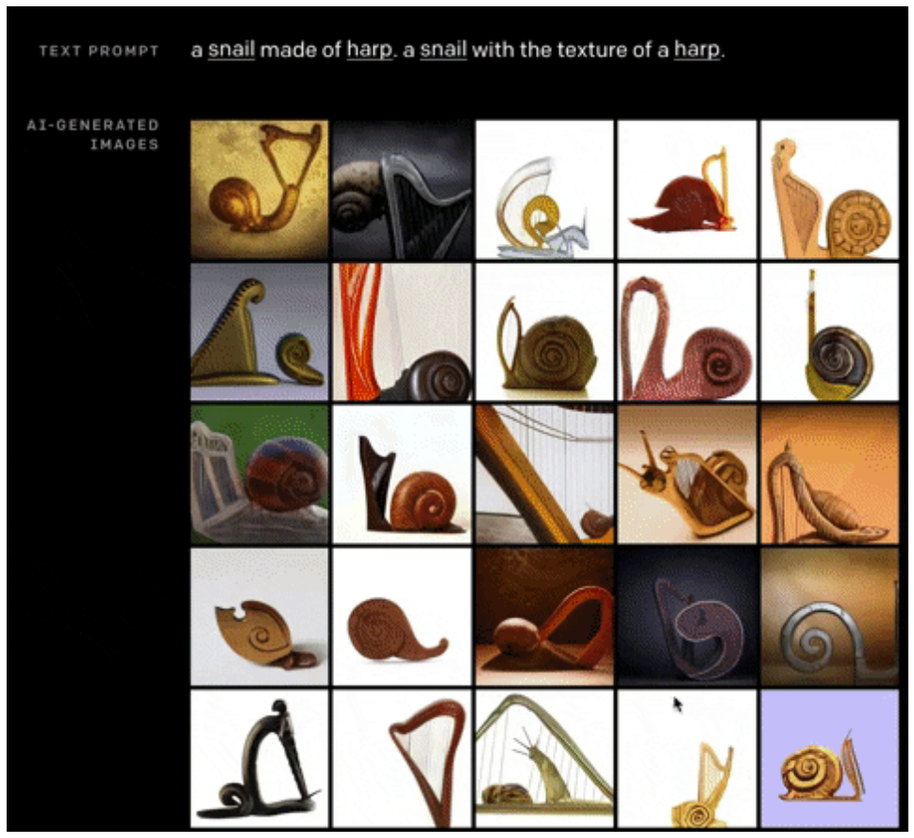
https://news.mit.edu/2019/intelligent-towing-tank-propels-research-1209

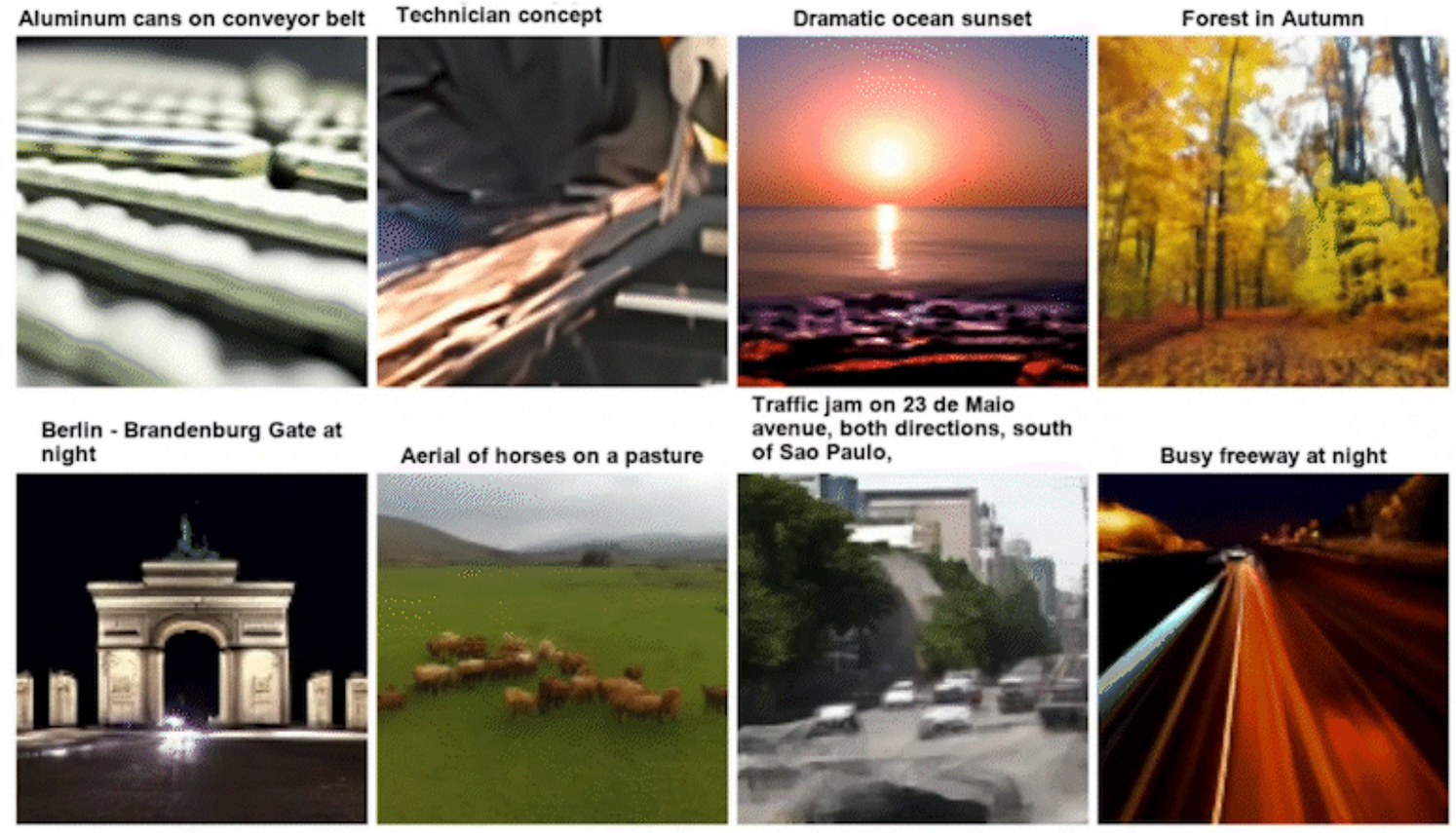
Recent success of machine learning

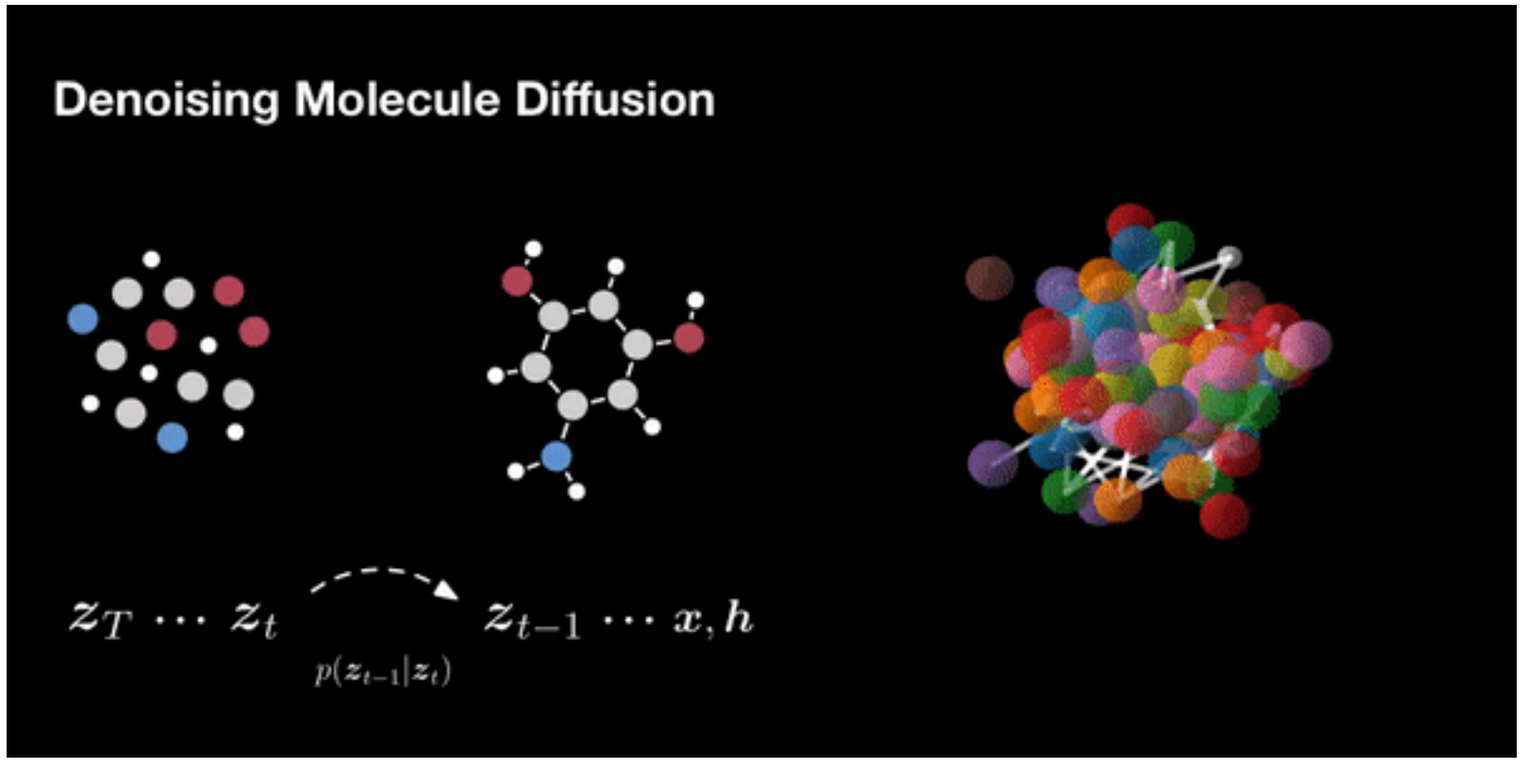




Working with high-dimensional data







Working with multiple data modalities

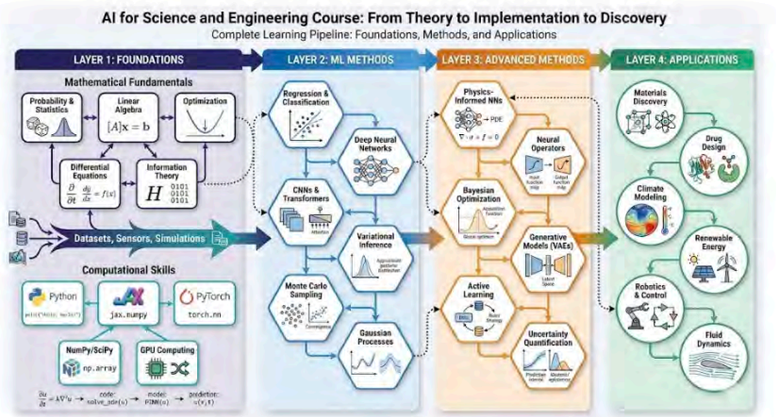
Gemini

Create a clean, modern academic infographic illustrating an AI for Science a...

Create a clean, modern academic infographic illustrating an AI for Science and Engineering course with complete learning pipeline.

LAYOUT: Four-layer hierarchical flow diagram (left to...

Show thinking (Nano Banana Pro)



Describe your image

Image

Thinking

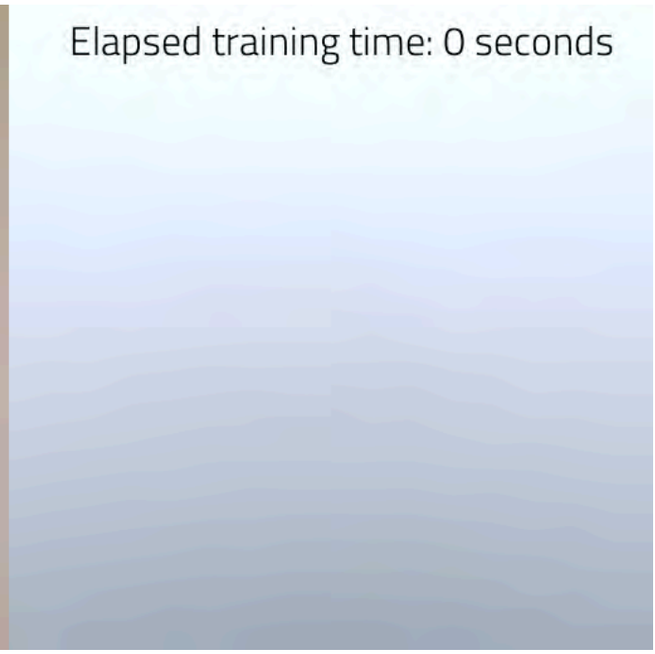
Data-driven science & engineering

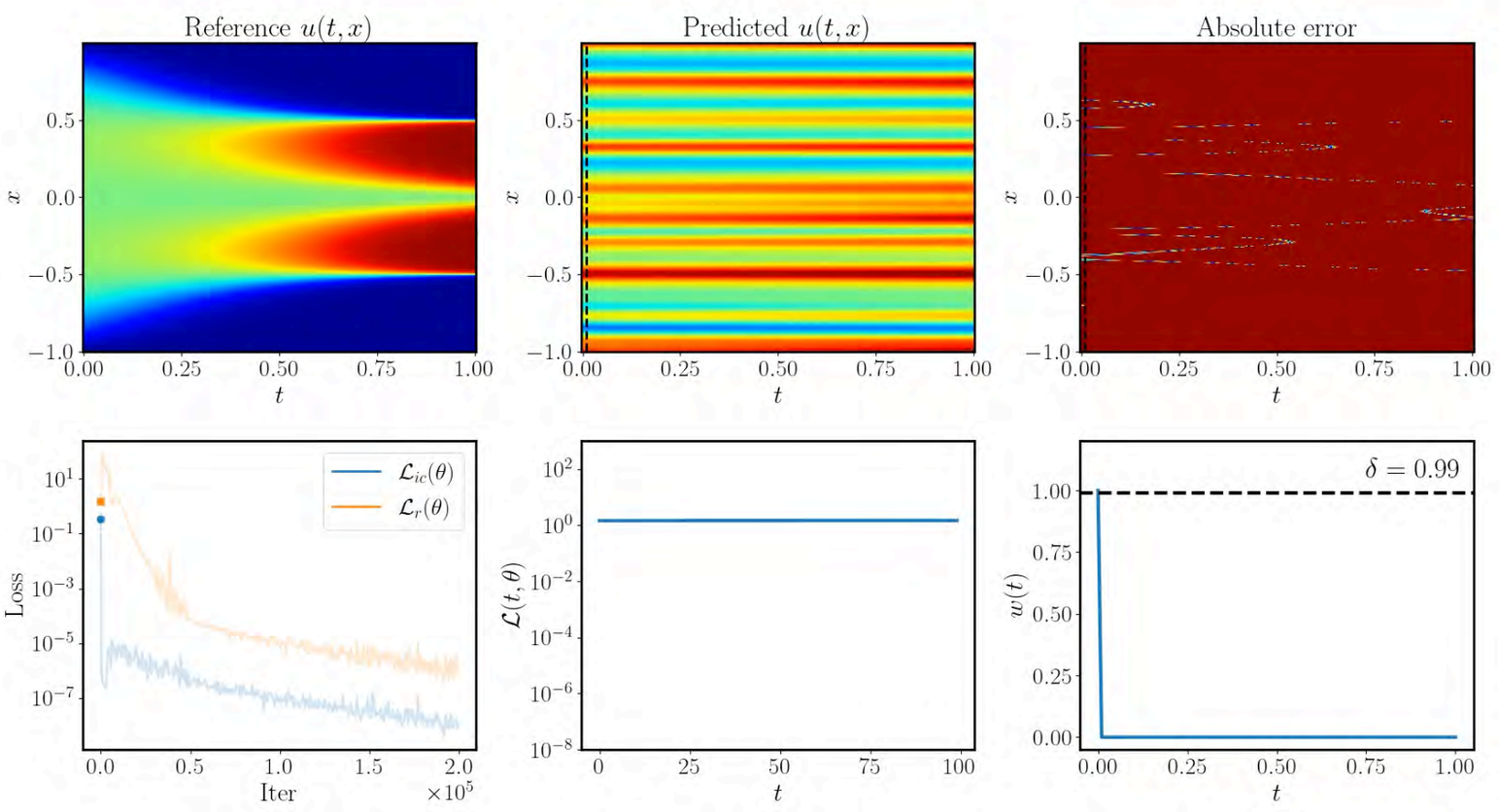
Neural gigapixel images

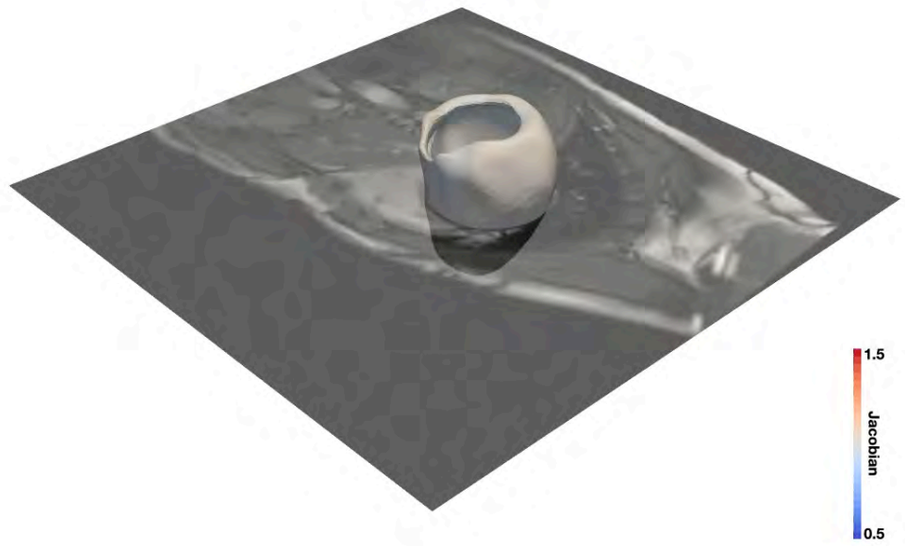
Neural SDF

NeRF

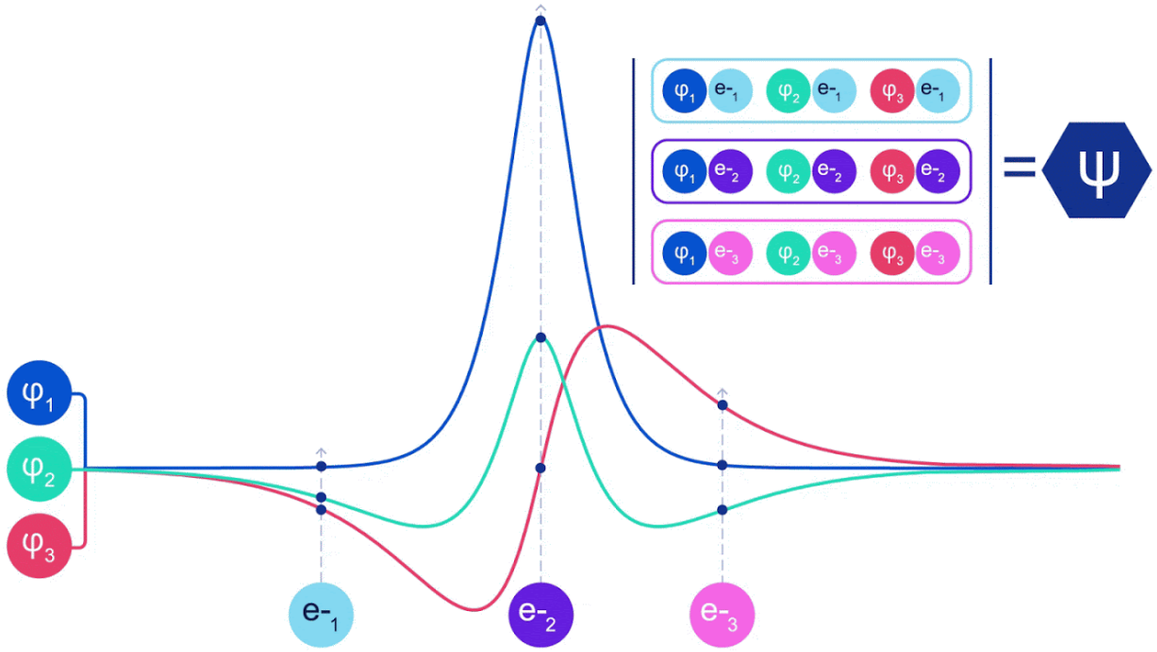
Neural volume

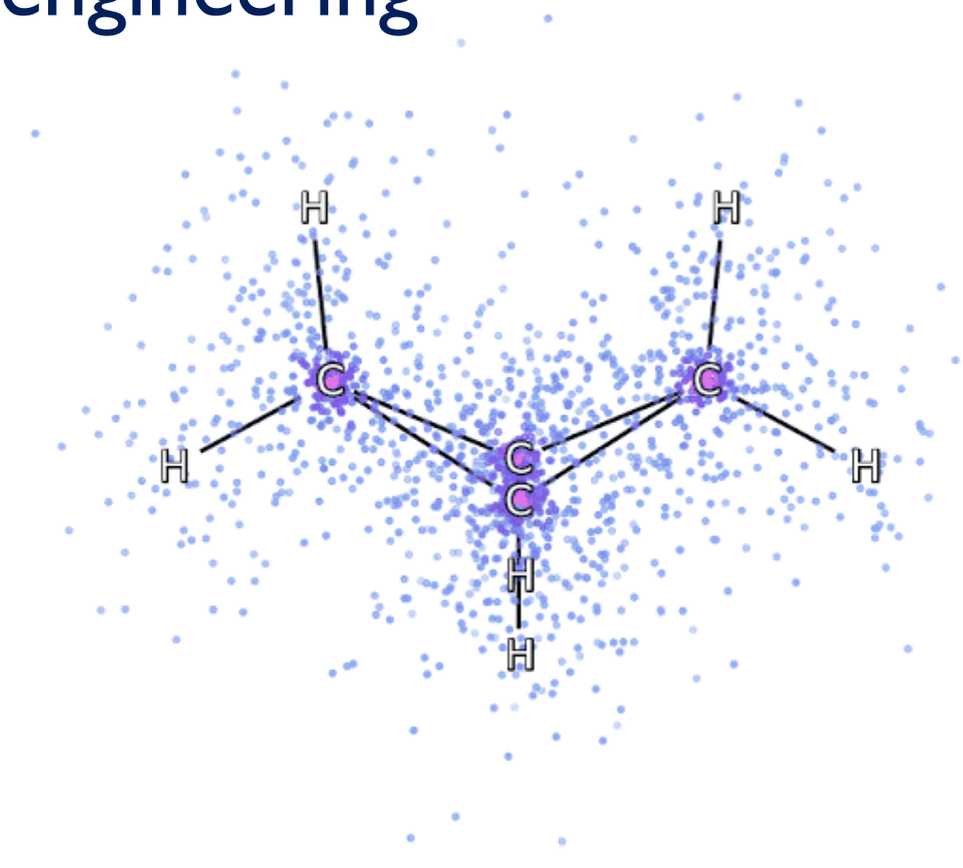


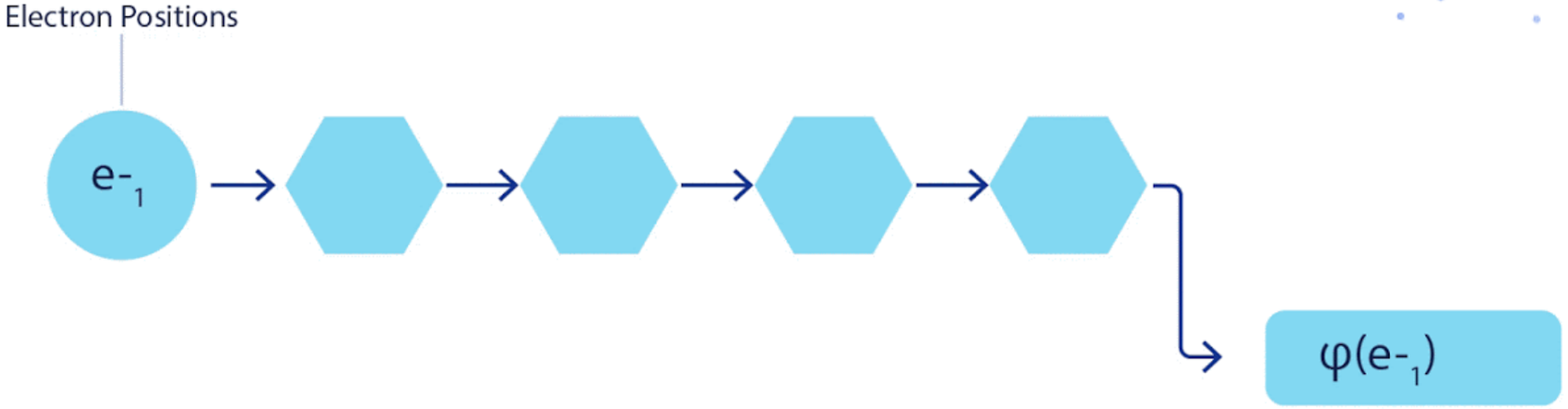




Data-driven science & engineering

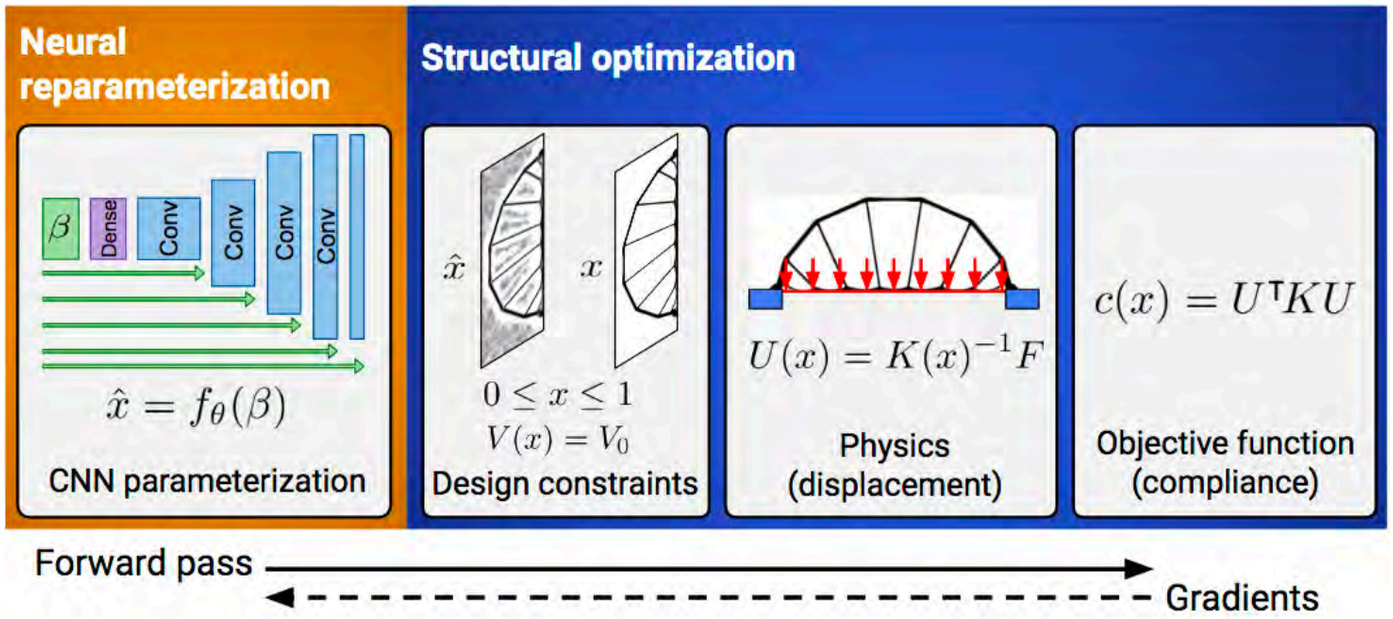


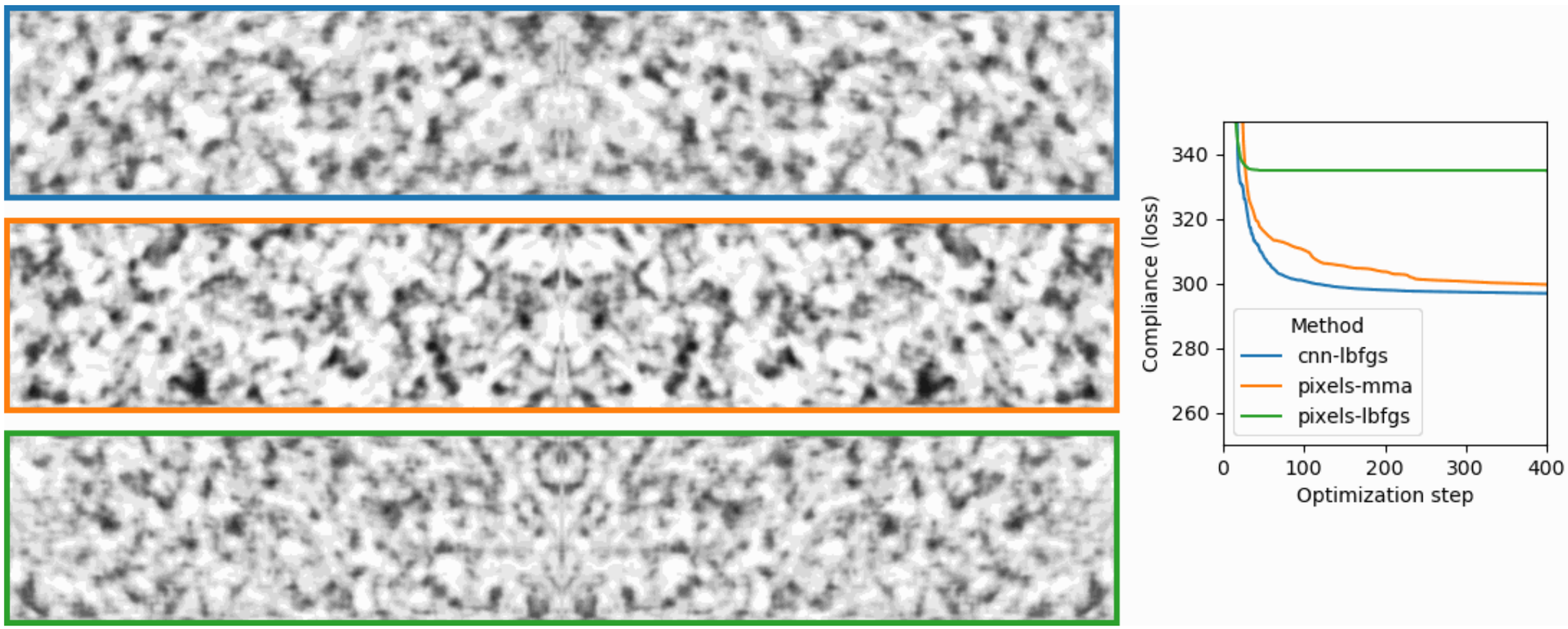




- [FermiNet: Quantum Physics and Chemistry from First Principles] https://deepmind.com/blog/article/FermiNet

Data-driven science & engineering

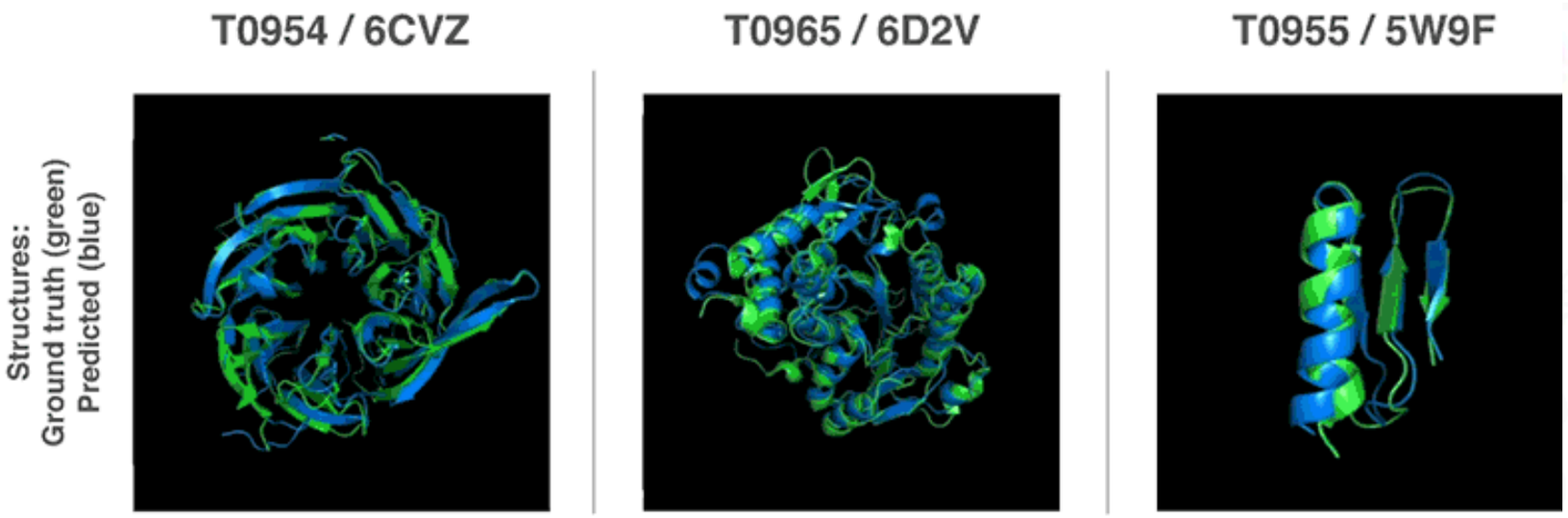


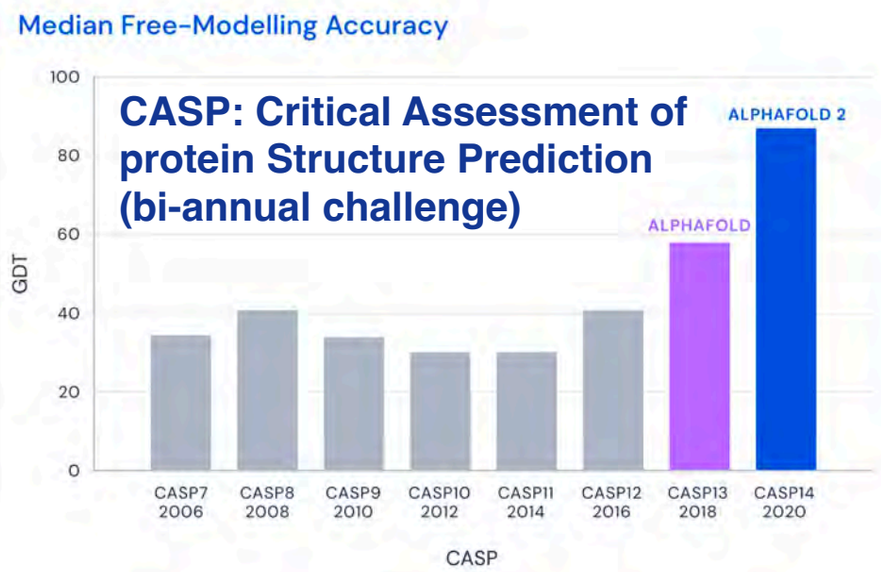


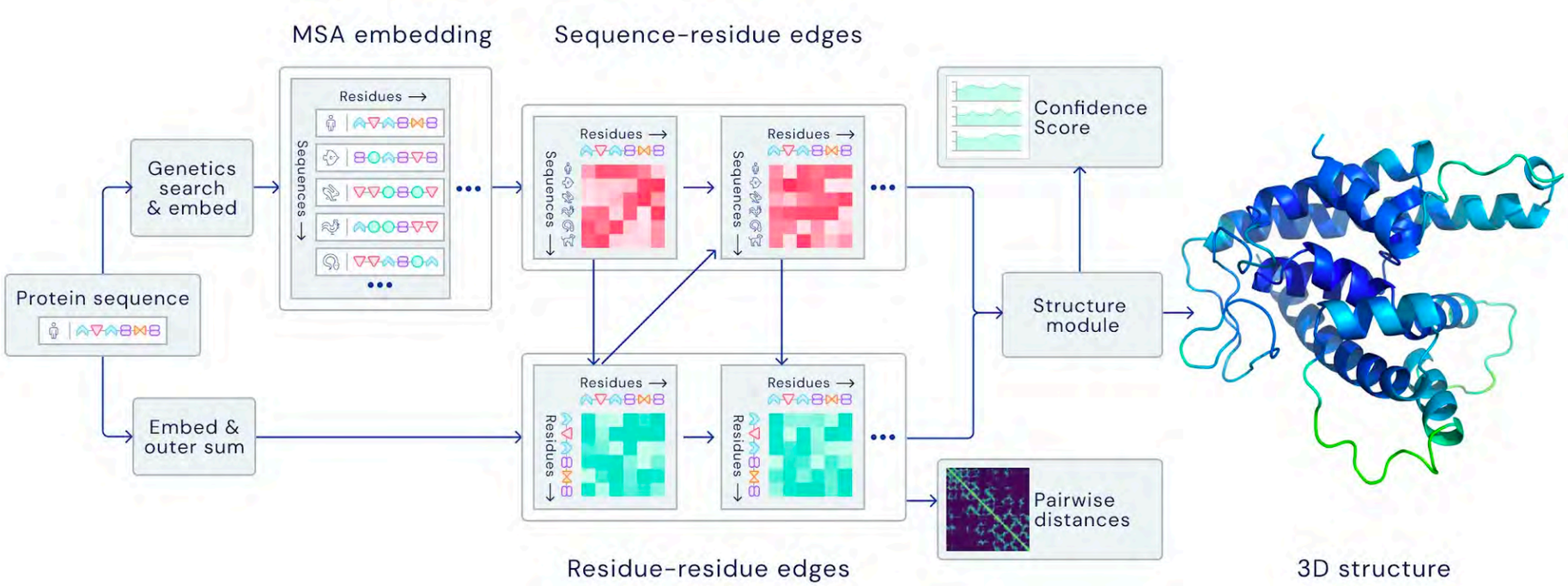
Optimizing a bridge structure. In the top frame, optimization happens in the weight space of a CNN. In the next two frames it happens on a finite element grid.

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

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.

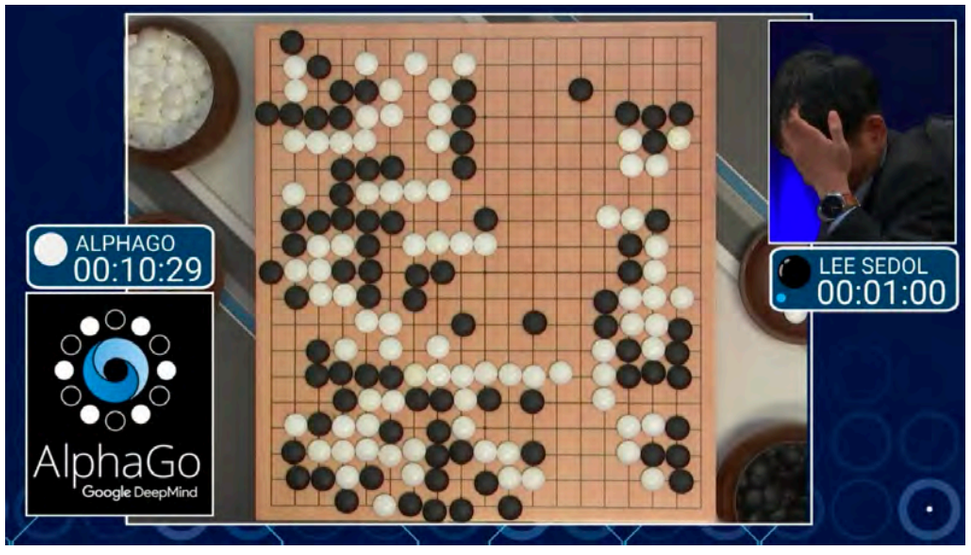
https://deepmind.com/blog/article/alphafold-a-solution-to-a-50-year-old-grand-challenge-in-biology

From predictions to decisions

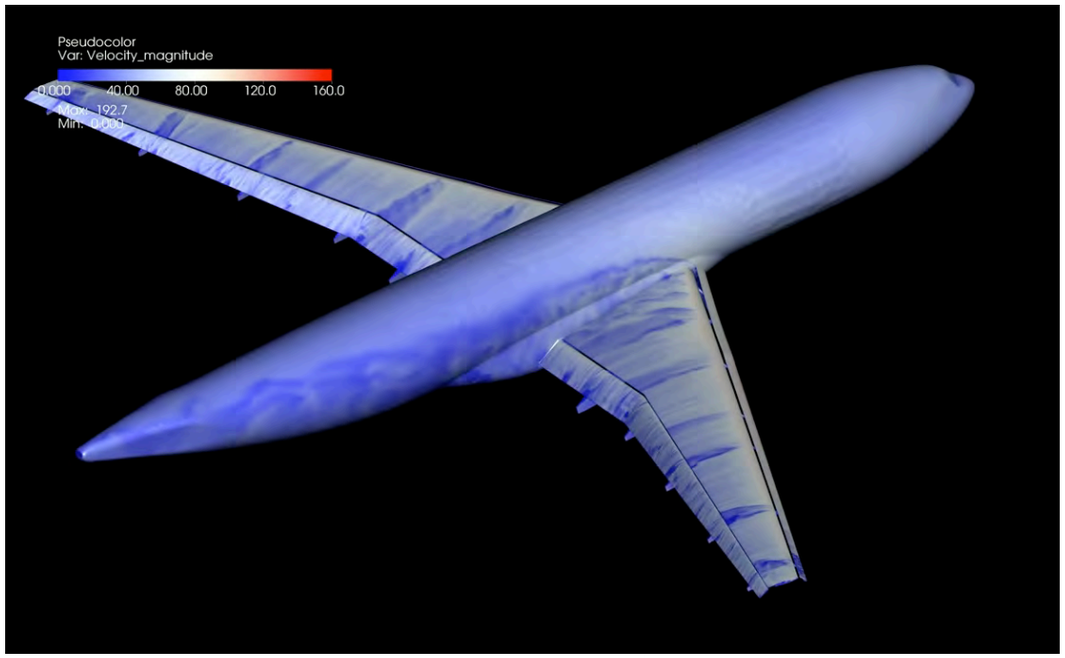
Google DeepMind's
Deep Q-learning

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

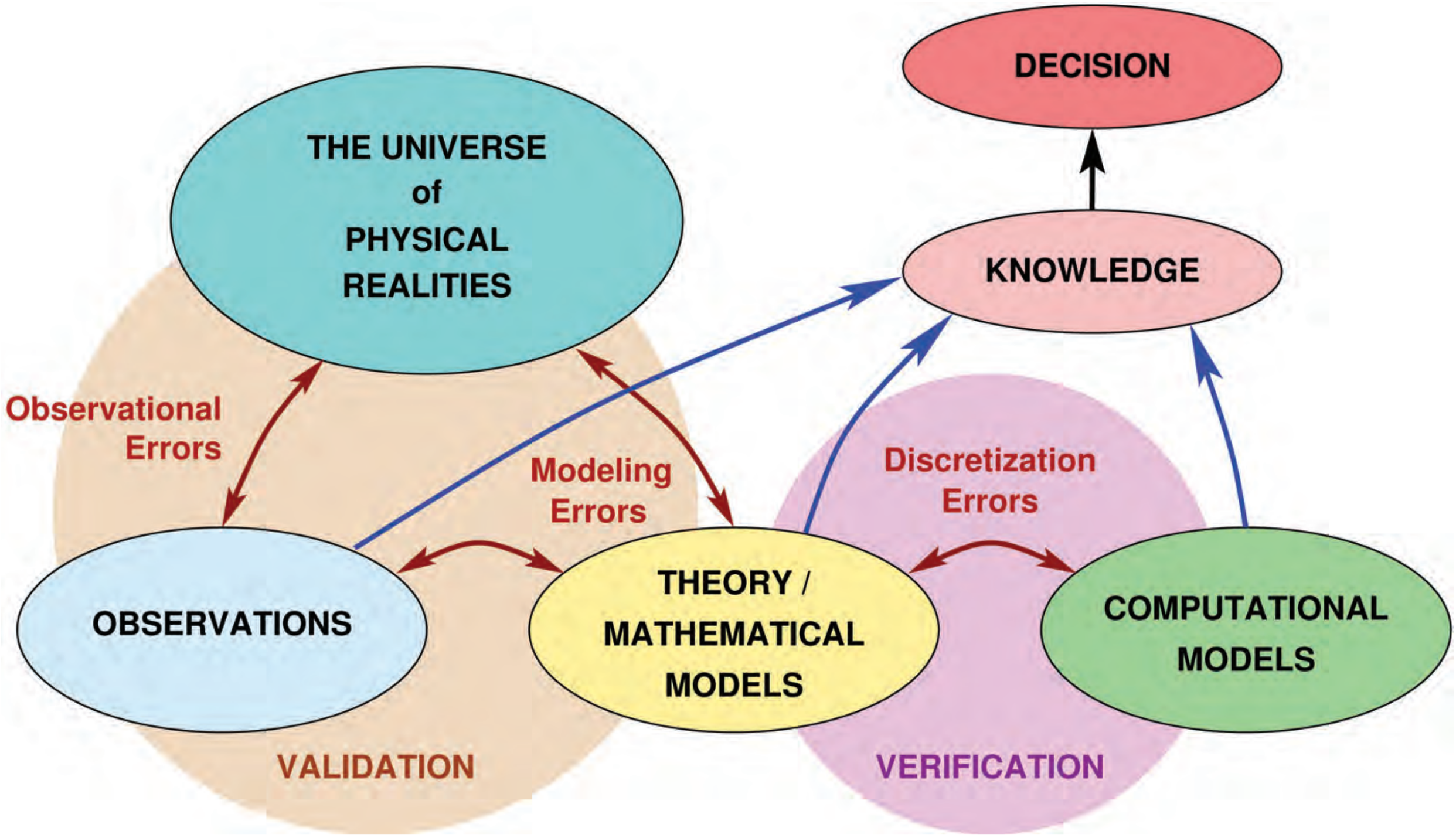
CSE

All models are wrong

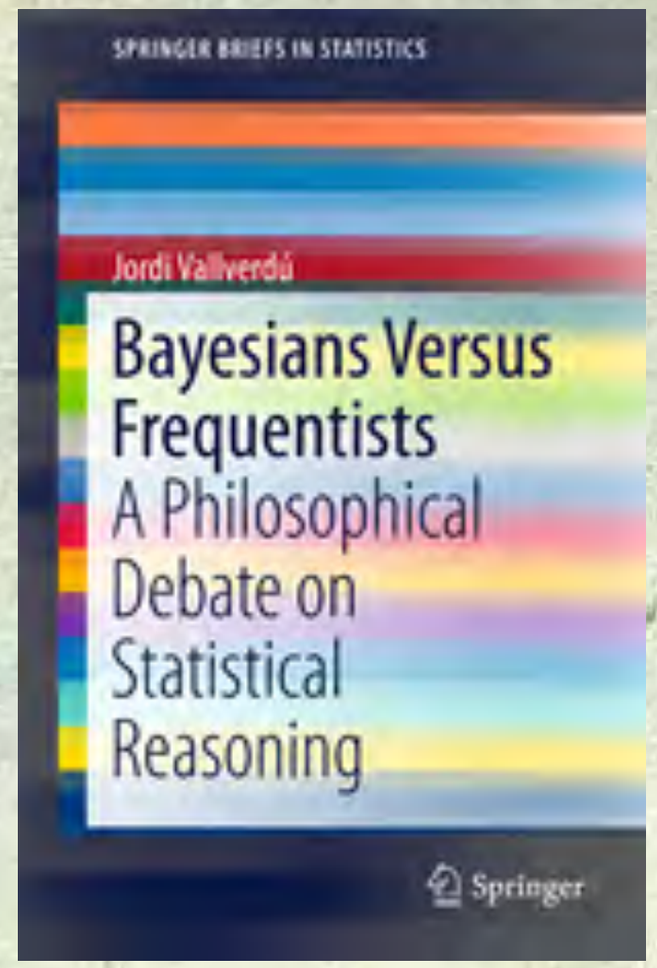
but some are useful

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

Foundational Competencies:

- Understand probabilistic modeling and apply Bayesian thinking to engineering problems
- Implement gradient-based optimization algorithms and understand stochastic training methods
- Quantify uncertainty in data-driven models and predictions for engineering applications

Deep Learning Expertise:

- Design and train neural networks (MLPs) for regression and classification tasks in science and engineering
- Apply convolutional neural networks (CNNs) to spatial data and image analysis
- Understand and implement Transformer architectures and attention mechanisms for sequence data
- Build recurrent neural networks (RNNs, LSTMs) for time-series and dynamical systems

Physics-Informed AI:

- Understand and implement physics-informed neural networks (PINNs) that incorporate physical constraints
- Apply neural operators to solve engineering problems governed by differential equations
- Integrate domain knowledge and conservation laws into machine learning models

Generative and Unsupervised Learning:

- Understand generative modeling approaches for data generation and compression
- Apply unsupervised learning techniques to discover patterns in engineering data

Practical Engineering Skills:

- Combine multiple ML techniques to solve complex engineering problems
- Design and execute computational experiments with proper validation strategies
- Evaluate trade-offs between traditional computational methods and AI approaches
- Implement end-to-end ML pipelines using modern Python frameworks
- Communicate technical results clearly to both technical and non-technical audiences

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).

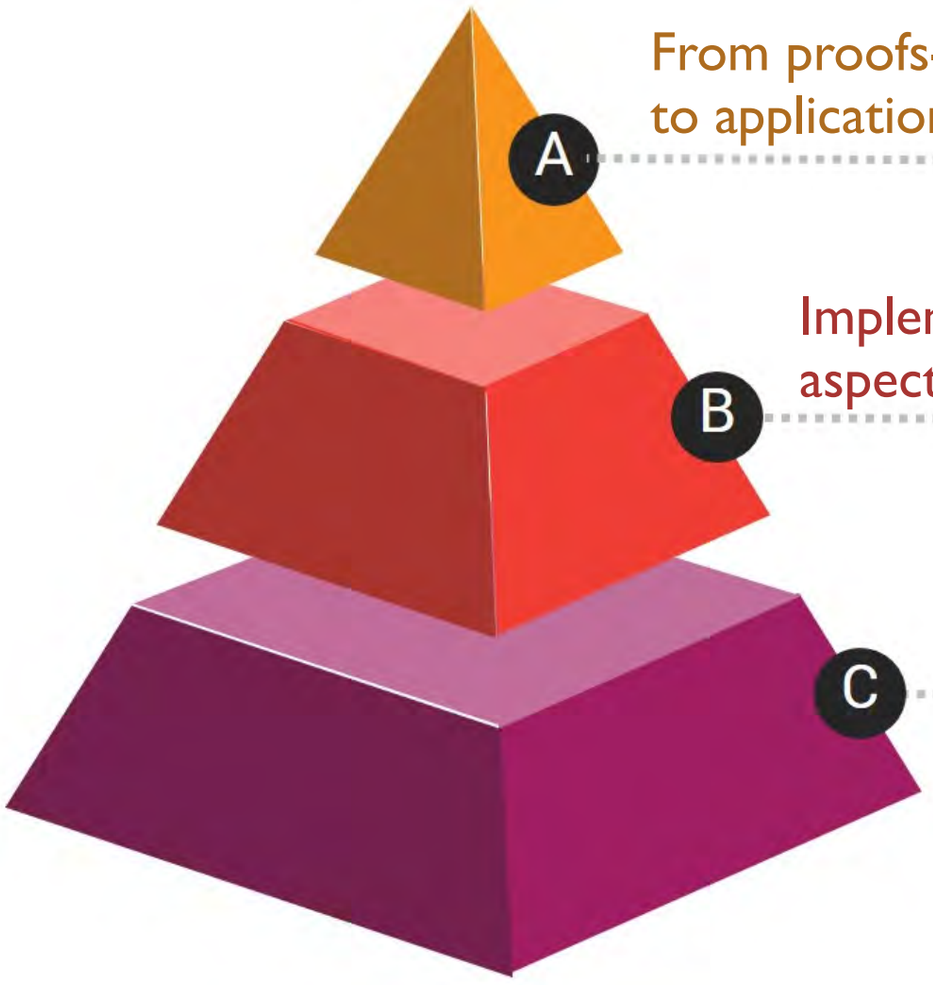
I am an experimentalist, what should I expect to learn?

(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%
Midterm exam	20%
Final project	40%
TOTAL	100%

computational proof-of-concept studies

mathematical foundations

applications

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

Supervised learning

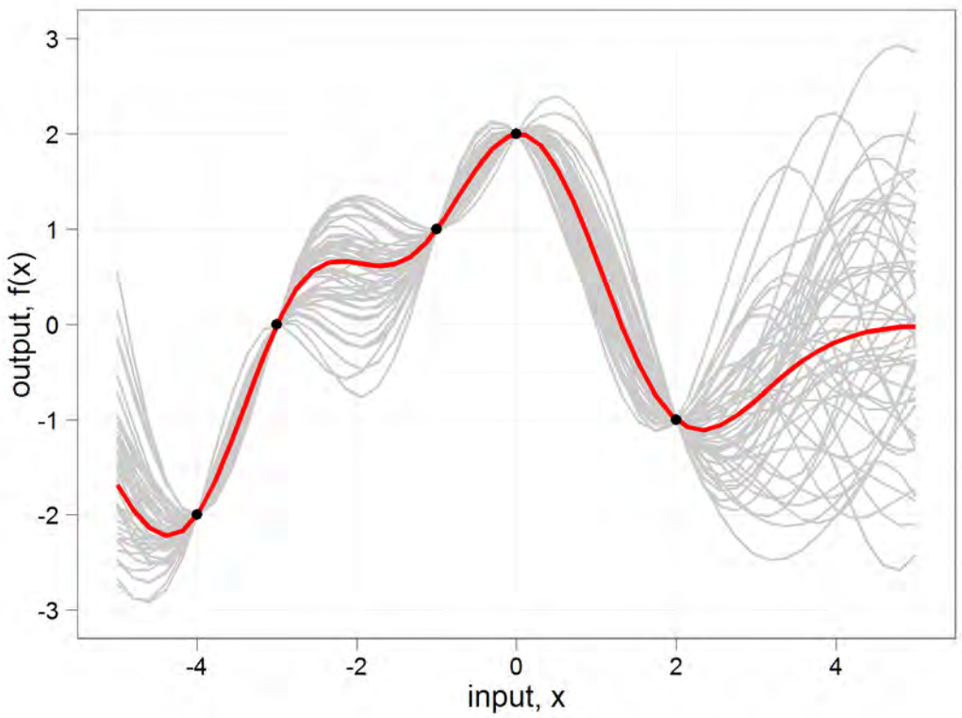
$$f : \mathcal{X} \to \mathcal{Y}$$

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

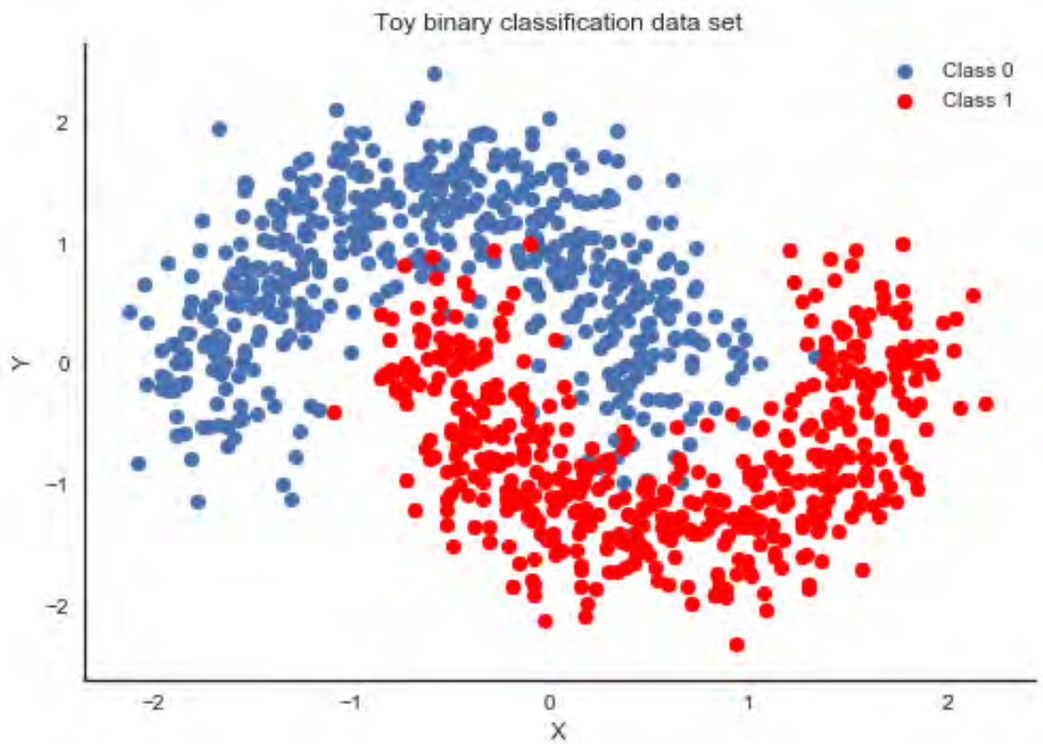
$$\boldsymbol{y} = f(\boldsymbol{x}) + \epsilon$$

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

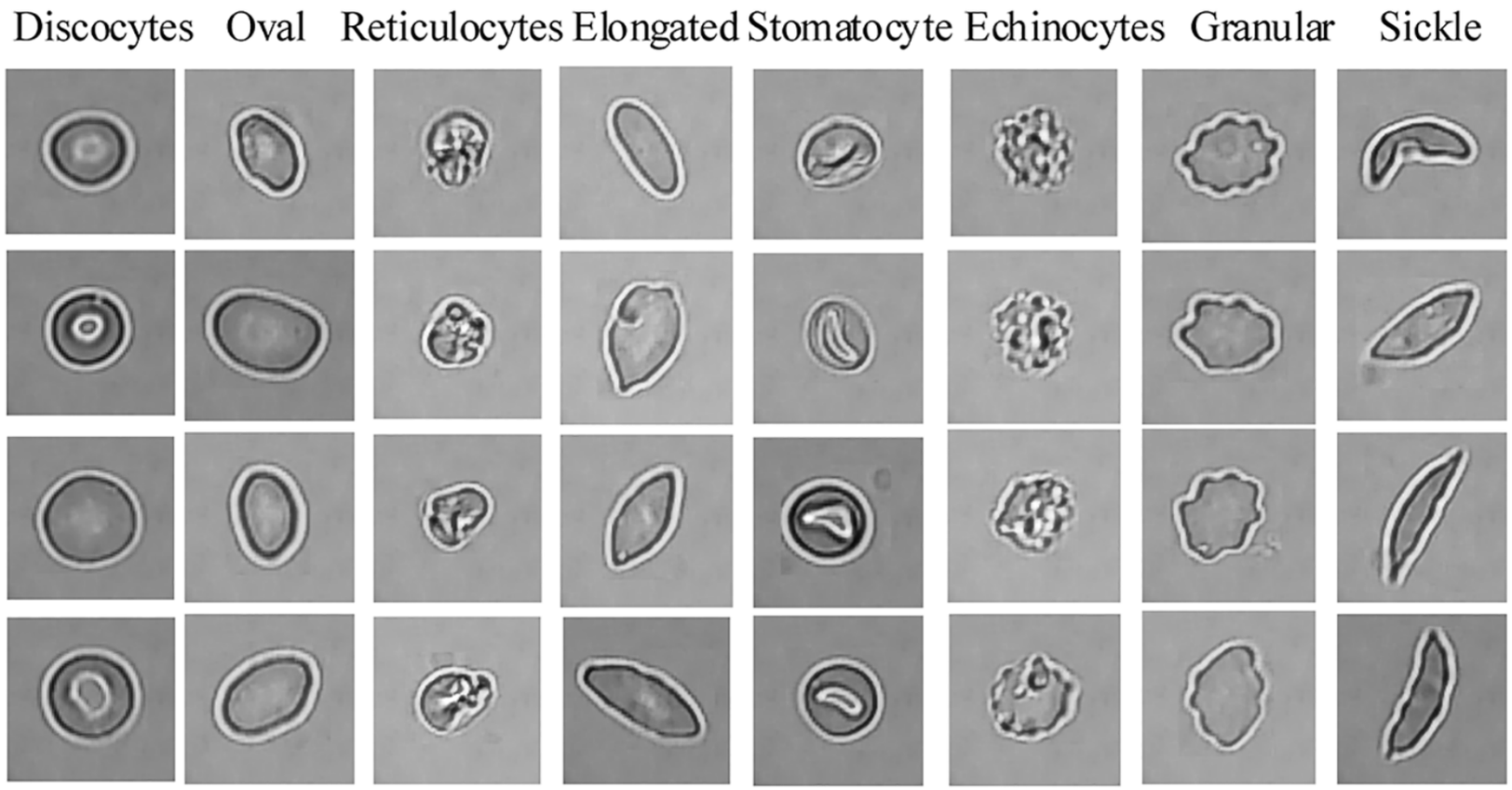
Regression



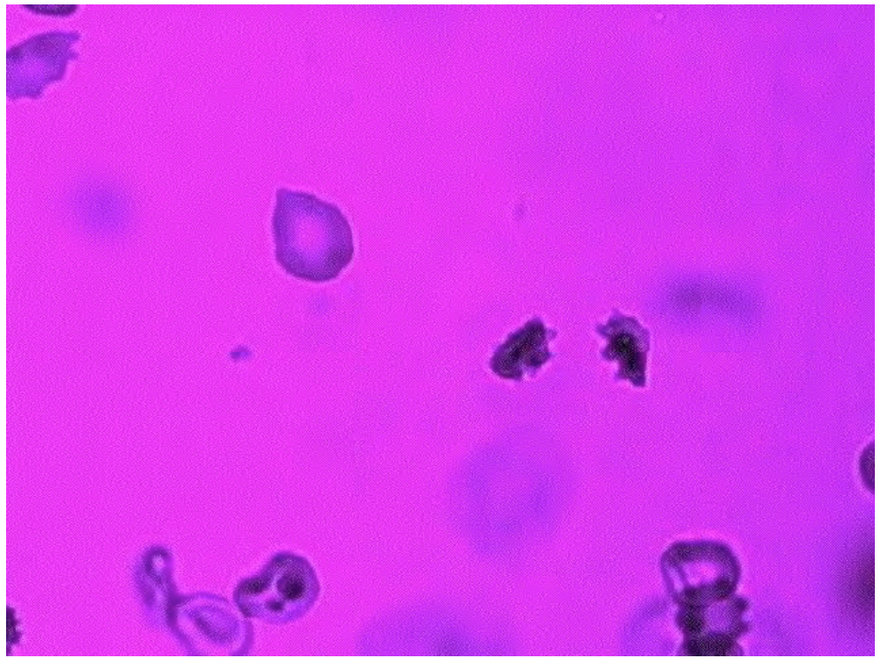
Classification



Classification



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



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

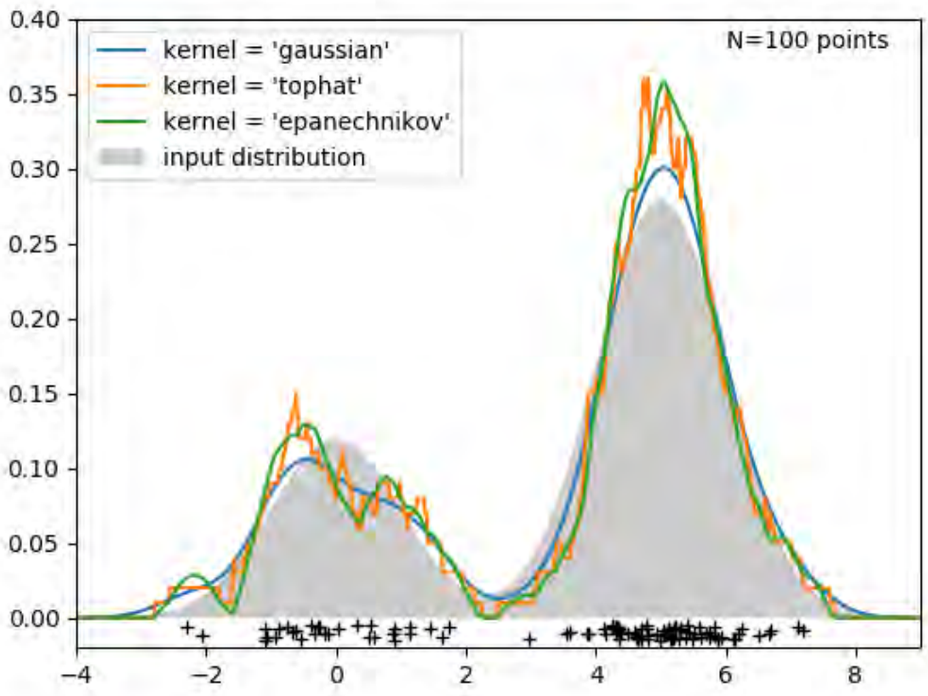
Unsupervised learning

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

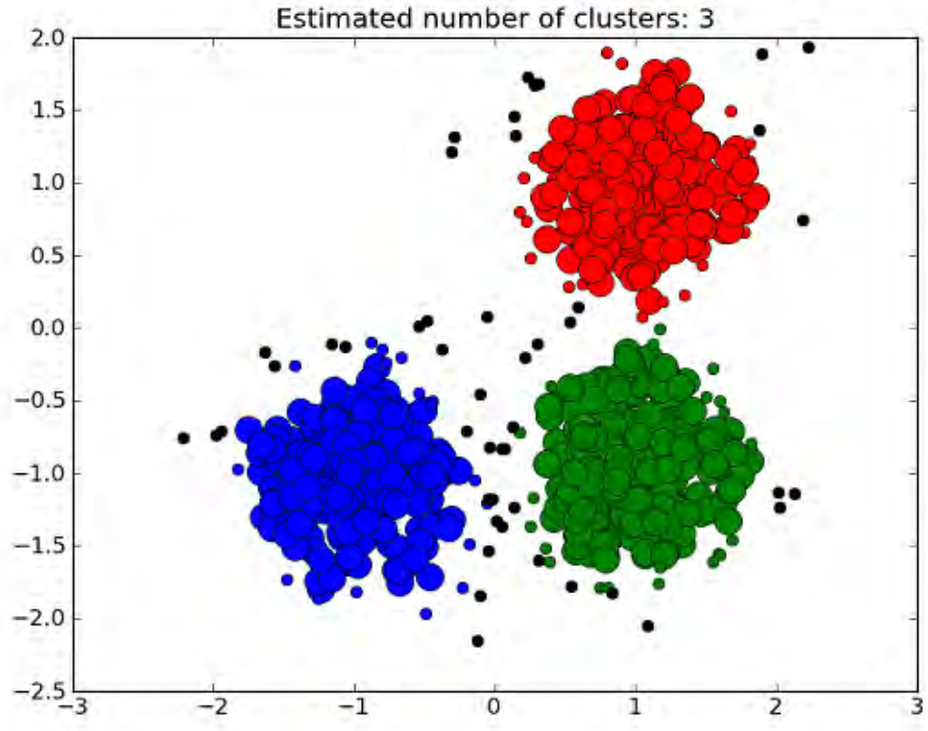
$$\boldsymbol{y} = f(\boldsymbol{z}) + \epsilon$$

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

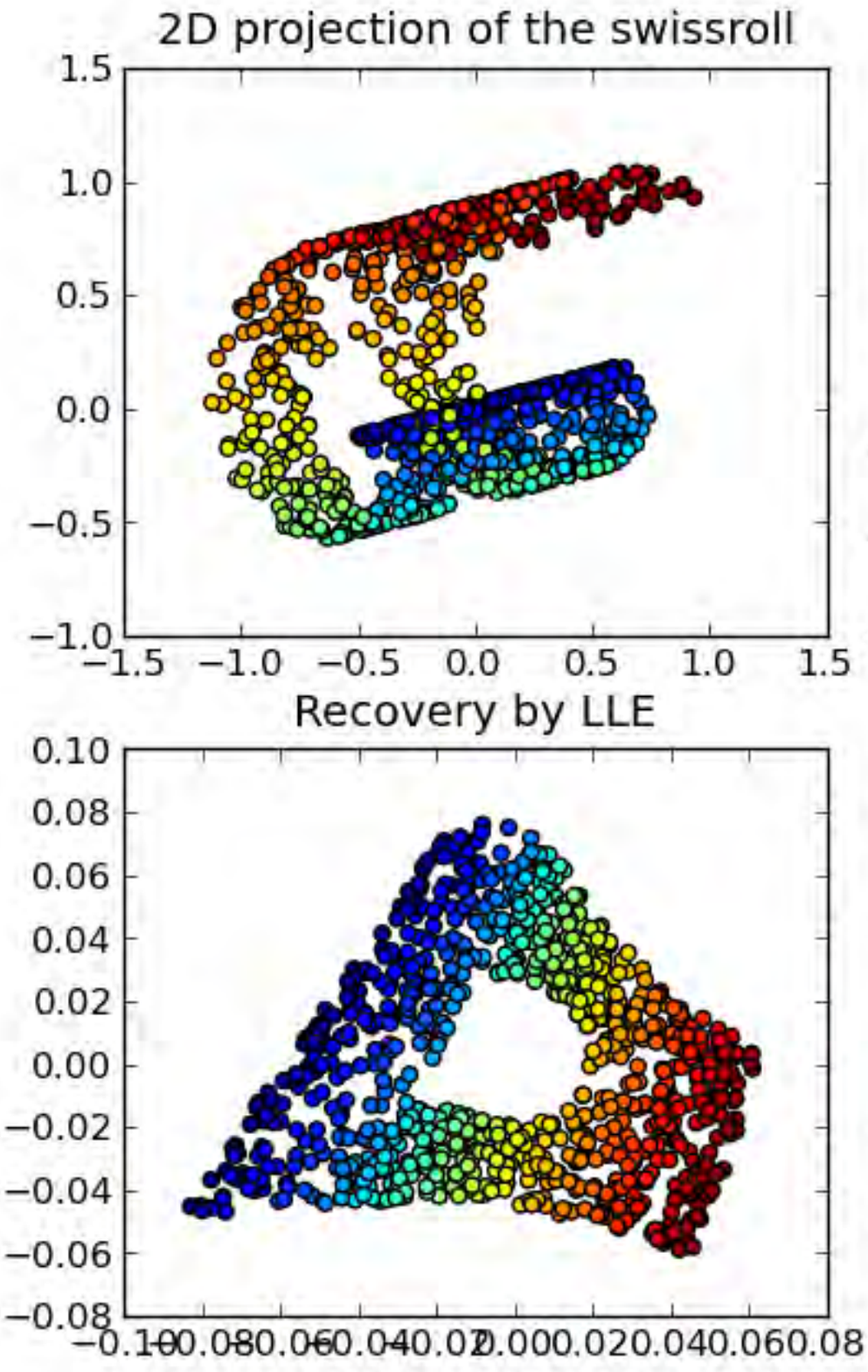
Density estimation

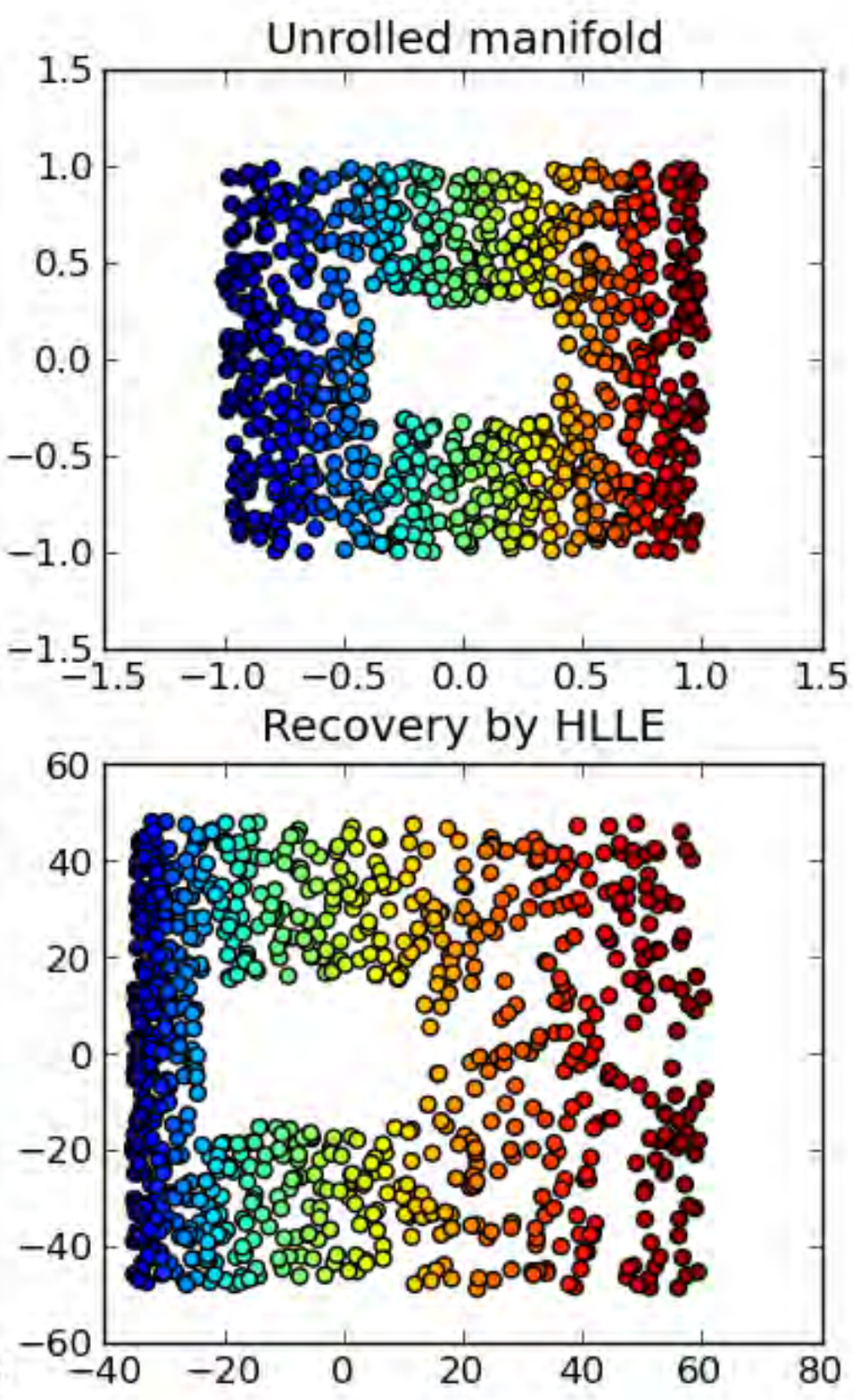


Clustering

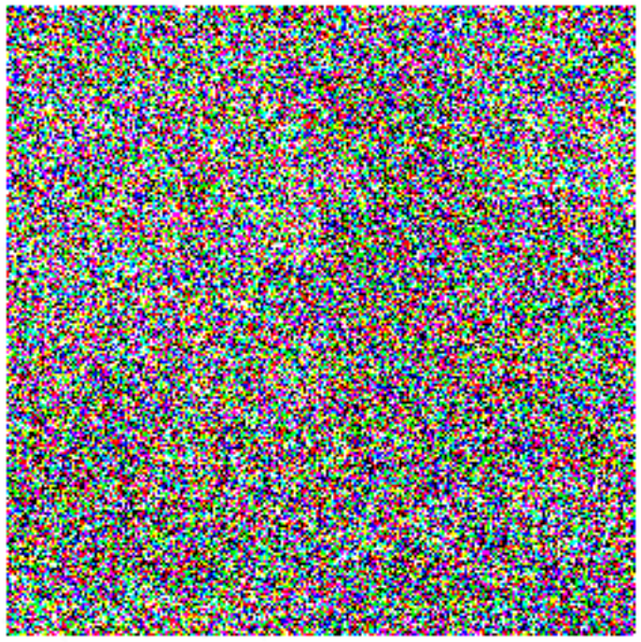


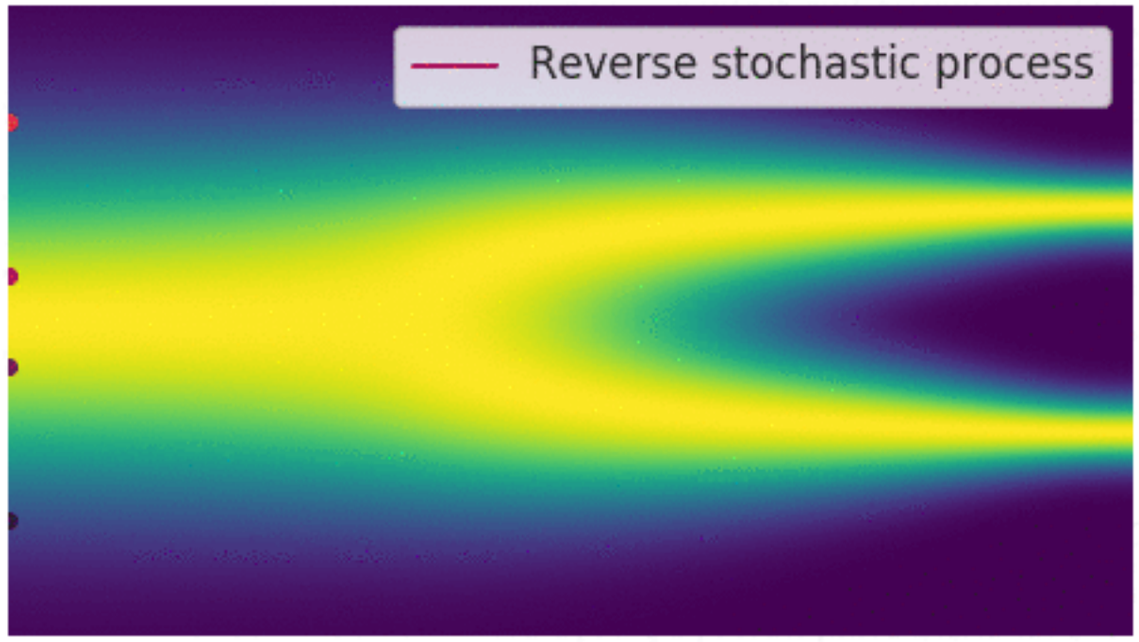
Dimensionality reduction

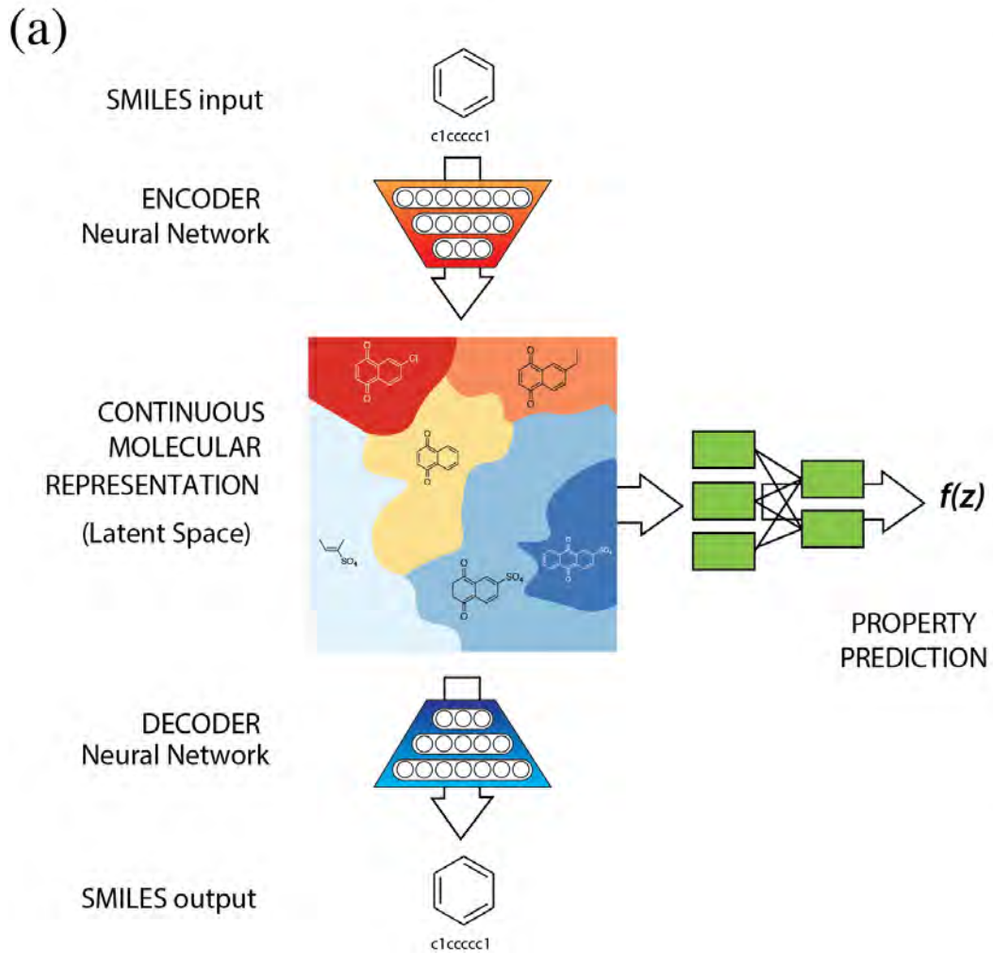


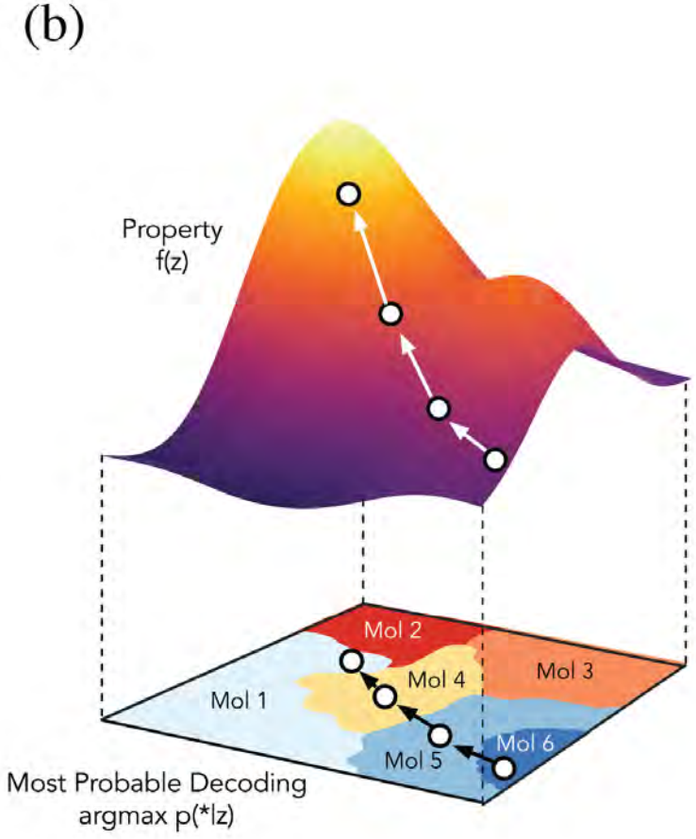


Generative Models









Course logistics

- Duration: 15 Weeks
- Time: Tuesdays and Thursdays, 1:45pm to 3:15pm (Towne 319)
- Dates: January 14 to April 29, 2026
- Instructor office hours: Tuesdays & Thursdays 3.30-4.30pm
- Weekly/bi-weekly HW assignments, mid-term exam, final project.

For more details visit the course website:
https://www.seas.upenn.edu/~meam4600/

Slides and code can be found on Github:
https://github.com/PredictiveIntelligenceLab/MEAM4600

Tentative Syllabus

Part I: Foundations

Week 1: Introduction to AI in Science and Engineering

Week 2: Fundamentals of Machine Learning for Scientific Computing

Week 3: Primer on Probability and Statistical Estimation

Week 4: Linear & Logistic Regression

Part II: Probabilistic Methods

Week 5: Variational Inference

Week 6: Monte Carlo Sampling

Week 7: Midterm Exam

Part III: Deep Learning Architectures

Week 8: Deep Learning Fundamentals: MLPs and CNNs

Week 9: Advanced Architectures: RNNs, GNNs, and Transformers

Part IV: Scientific Machine Learning

Week 10: Kernel Methods & Gaussian Process Regression

Week 11: Physics-Informed Neural Networks & Neural Operators

Week 12: Bayesian Optimization and Active Learning

Part V: Unsupervised and Generative Methods

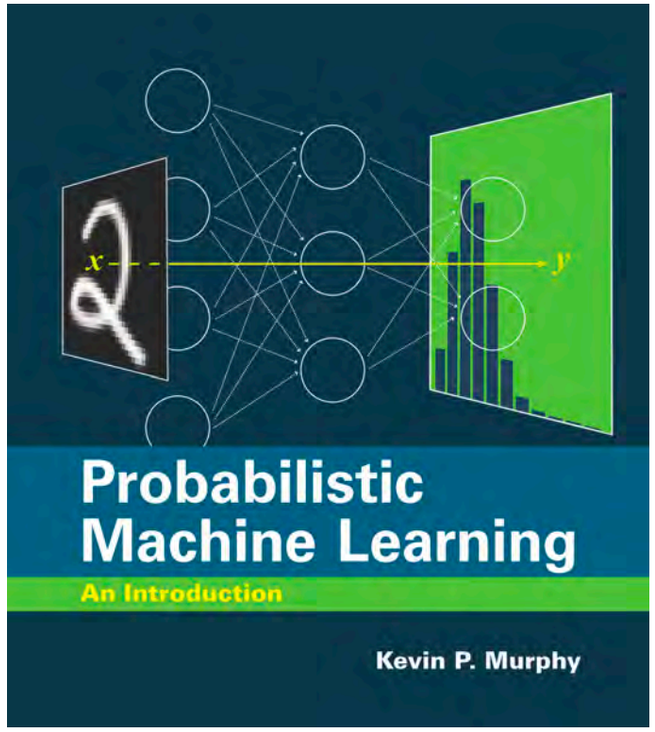
Week 13: Unsupervised Learning: PCA and VAEs

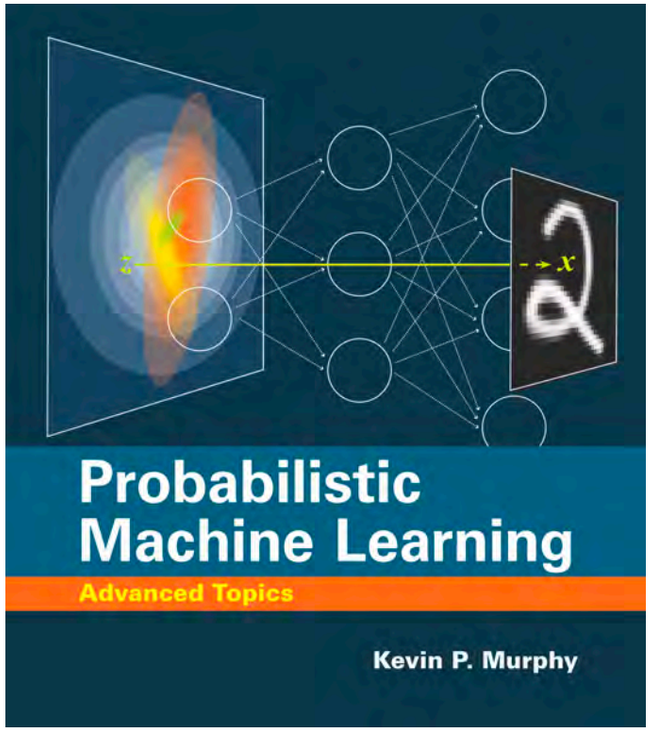
Week 14: Generative Models

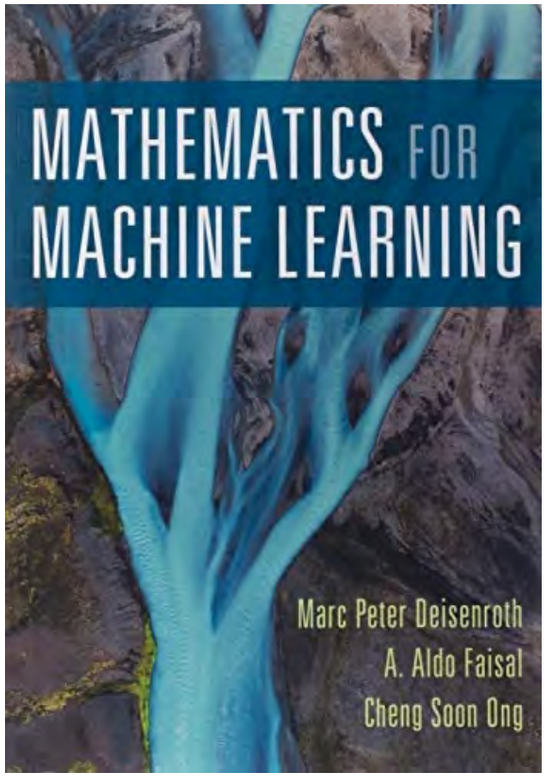
Week 15: Final Project Presentations

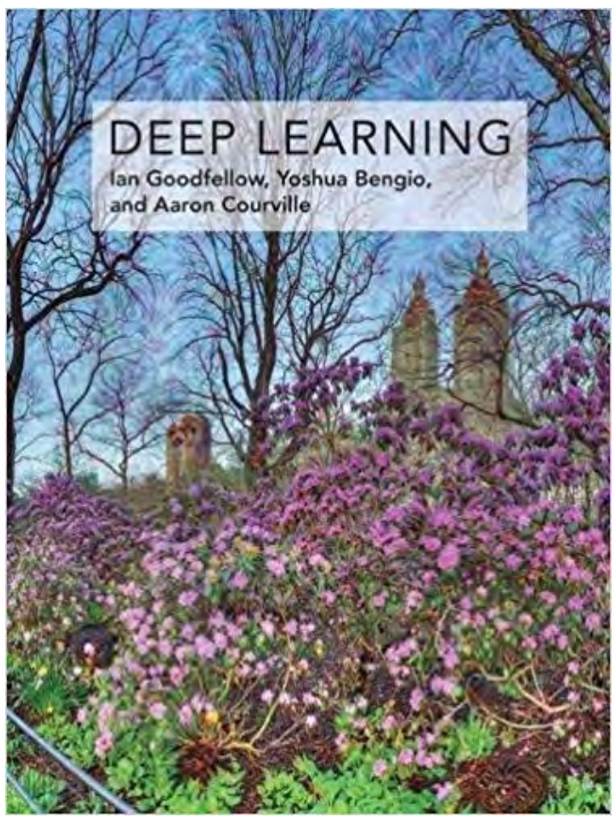
Note: Each week includes lecture content, hands-on coding exercises, and real-world engineering examples. Homework assignments are distributed throughout the semester to reinforce concepts progressively.

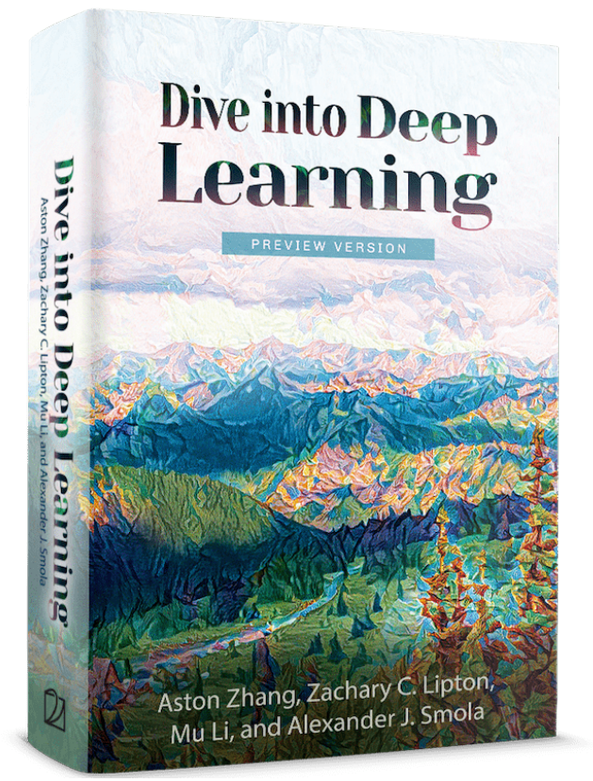
Textbooks

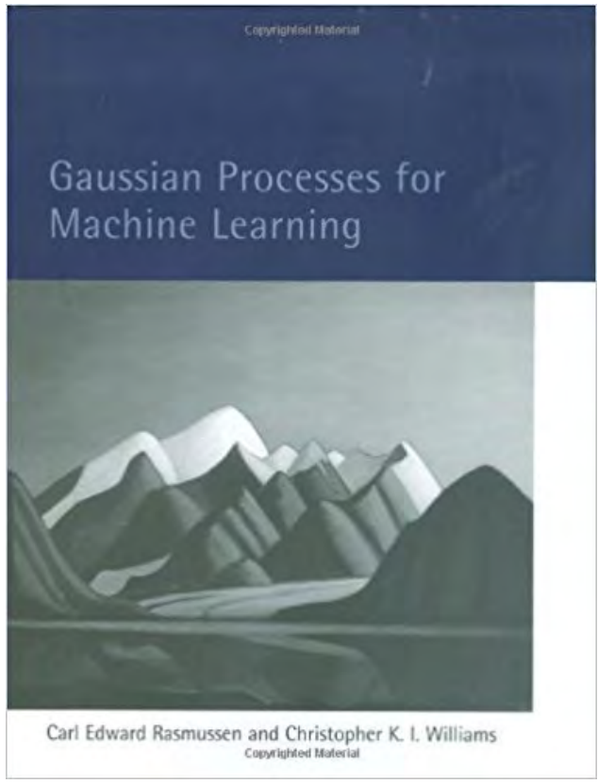

Probabilistic
Machine Learning
An Introduction
Kevin P. Murphy


Probabilistic
Machine Learning
Advanced Topics
Kevin P. Murphy


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

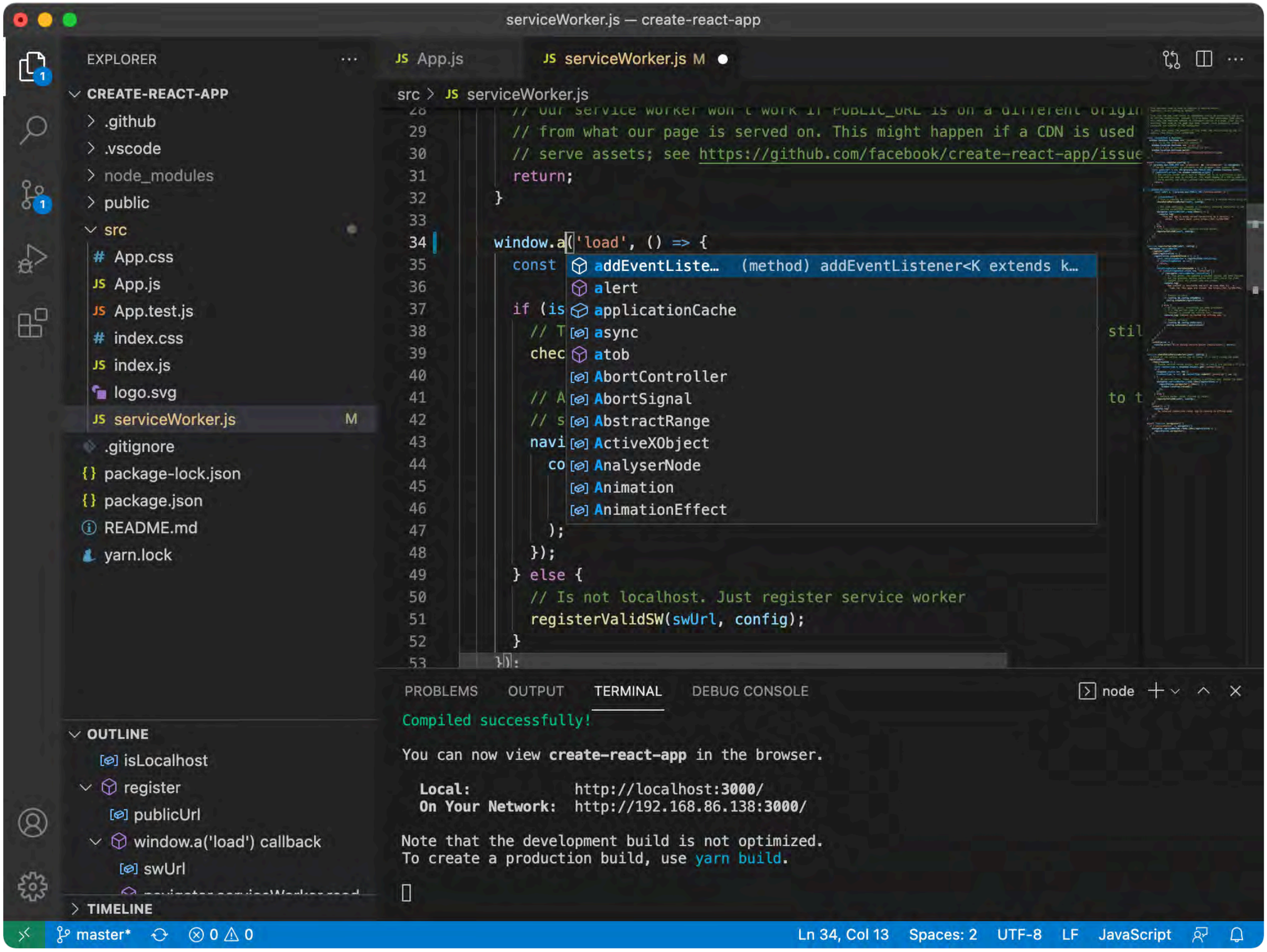
- Introduction
- Syllabus review
- Presentation of diverse applications that showcase the use of data, modeling, and scientific computation.
- Overview of the main themes and goals of this course
- Primer on Probability and Statistics

Reading

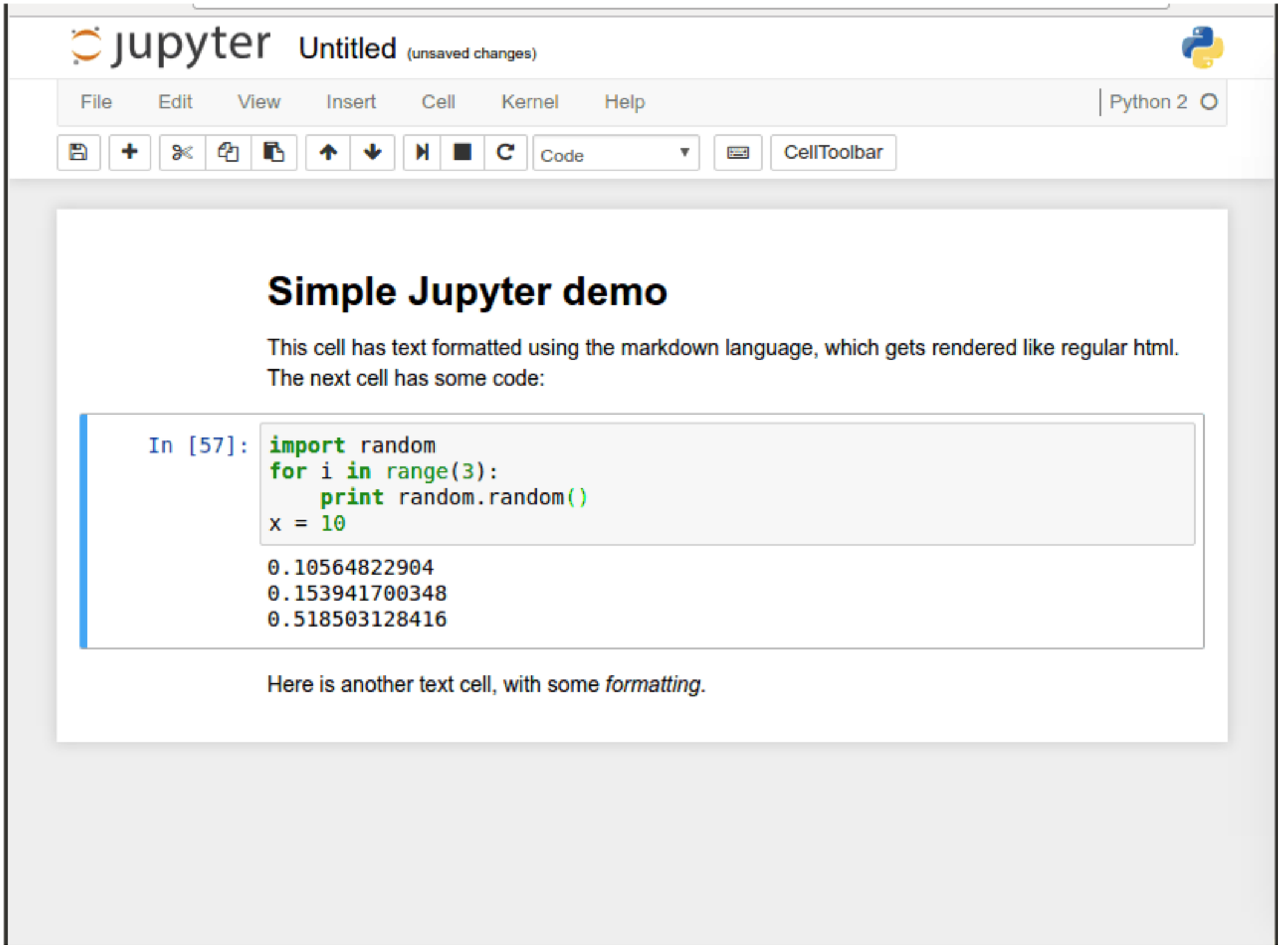
- The Emergence of Predictive Computational Science:
  - Computer predictions with quantified uncertainty (Oden, Moser, & Ghattas, 2010)
  - Lecture by J.T Oden.
- Review papers on recent advances in machine learning:
  - Probabilistic machine learning and artificial intelligence (Ghahramani, 2015)
  - 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.

VSCode


serviceWorker.js — create-react-app
EXPLORER
CREATE-REACT-APP
.github
.vscode
node_modules
public
src
App.css
App.js
App.test.js
index.css
index.js
logo.svg
serviceWorker.js M
.gitignore
package-lock.json
package.json
README.md
yarn.lock
App.js
serviceWorker.js M
src > serviceWorker.js
// from what our page is served on. This might happen if a CDN is used
// serve assets; see https://github.com/facebook/create-react-app/issue
return;
}
window.a('load', () => {
const
addEventListe… (method) addEventListener<K extends k…
alert
applicationCache
async
atob
AbortController
AbortSignal
AbstractRange
ActiveXObject
AnalyserNode
Animation
AnimationEffect
);
});
} else {
// Is not localhost. Just register service worker
registerValidSW(swUrl, config);
}
OUTLINE
isLocalhost
register
publicUrl
window.a('load') callback
swUrl
TIMELINE
PROBLEMS
OUTPUT
TERMINAL
DEBUG CONSOLE
node
Compiled successfully!
You can now view create-react-app in the browser.
Local: http://localhost:3000/
On Your Network: http://192.168.86.138:3000/
Note that the development build is not optimized.
To create a production build, use yarn build.
master*
0 0
Ln 34, Col 13
Spaces: 2
UTF-8
LF
JavaScript

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