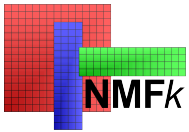
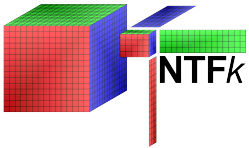
Unsupervised and Physics-Informed Machine Learning of Big and Noisy Data

Velimir V. Vesselinov (monty) (vvv@lanl.gov)
Earth and Environmental Sciences Division
Los Alamos National Laboratory, NM, USA
http://tensors.lanl.gov





Machine Learning (ML) methods

- **Supervised** ML: learns everything from data
  ⇒ requires big training datasets
  ⇒ highly impacted by noise

- **Physics-informed** ML: learns from data but includes preconceived knowledge about the governing processes
  ⇒ requires smaller training datasets
  ⇒ produces better predictability with lower uncertainty
  ⇒ robust to data noise

- **Unsupervised** ML: extracts features from data that can be applied for categorization and prediction
  ⇒ unbiased analyses not impacted by data labeling and physics assumptions

Machine Learning (ML) methods

- **Supervised** ML: learns everything from data
  $\Rightarrow$ requires big training datasets
  $\Rightarrow$ highly impacted by noise

- **Physics-informed** ML: learns from data but includes preconceived knowledge about the governing processes
  $\Rightarrow$ requires smaller training datasets
  $\Rightarrow$ produces better predictability with lower uncertainty
  $\Rightarrow$ robust to data noise

- **Unsupervised** ML: extracts features from data that can be applied for categorization and prediction
  $\Rightarrow$ unbiased analyses not impacted by data labeling and physics assumptions

Machine Learning (ML) methods

- **Supervised** ML: learns everything from data
  ⇒ requires big training datasets
  ⇒ highly impacted by noise

- **Physics-informed** ML: learns from data but includes preconceived knowledge about the governing processes
  ⇒ requires smaller training datasets
  ⇒ produces better predictability with lower uncertainty
  ⇒ robust to data noise

- **Unsupervised** ML: extracts features from data that can be applied for categorization and prediction
  ⇒ unbiased analyses not impacted by data labeling and physics assumptions

Why unsupervised Machine Learning (ML)?

- **Supervised** ML: requires “labeling” (prior categorization (knowledge) about the processed data)

  Example: Recognizes images of cats and dogs after extensive training; but cannot recognize horses if not trained

  Cannot discover something that we do not know already

- **Unsupervised** ML: extracts hidden features (signals) in the processed data without any prior information (**exploratory analysis**; **data-driven science**)

  Example: Identifies features that distinguish images of animals (e.g., cats, dogs, horses, etc.)

Why unsupervised Machine Learning (ML)?

- **Supervised** ML: requires “labeling” (prior categorization (knowledge) about the processed data)

  Example: Recognizes images of cats and dogs after extensive training; but cannot recognize horses if not trained

  Cannot discover something that we do not know already

- **Unsupervised** ML: extracts hidden features (signals) in the processed data without any prior information (**exploratory analysis**; **data-driven science**)

  Example: Identifies features that distinguish images of animals (e.g., cats, dogs, horses, etc.)

Why unsupervised Machine Learning (ML)?

- **Supervised** ML: requires “labeling” (prior categorization (knowledge) about the processed data)

  Example: Recognizes images of cats and dogs after extensive training; but cannot recognize horses if not trained

  Cannot discover something that we do not know already

- **Unsupervised** ML: extracts hidden features (signals) in the processed data without any prior information (**exploratory analysis**; **data-driven science**)

  Example: Identifies features that distinguish images of animals (e.g., cats, dogs, horses, etc.)

Why unsupervised Machine Learning (ML)?

- **Supervised** ML: requires “labeling” (prior categorization (knowledge) about the processed data)

  Example: Recognizes images of cats and dogs after extensive training; but cannot recognize horses if not trained

  Cannot discover something that we do not know already

- **Unsupervised** ML: extracts hidden features (signals) in the processed data without any prior information (**exploratory analysis**; **data-driven science**)

  Example: Identifies features that distinguish images of animals (e.g., cats, dogs, horses, etc.)

Why unsupervised Machine Learning (ML)?

- **Supervised** ML: requires “labeling” (prior categorization (knowledge) about the processed data)

  Example: Recognizes images of cats and dogs after extensive training; but cannot recognize horses if not trained

  Cannot discover something that we do not know already

- **Unsupervised** ML: extracts hidden features (signals) in the processed data without any prior information (**exploratory analysis**; **data-driven science**)

  Example: Identifies features that distinguish images of animals (e.g., cats, dogs, horses, etc.)

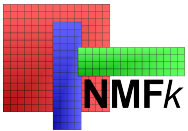
Unsupervised Machine Learning Applications: Data Analytics / Model Diagnostics

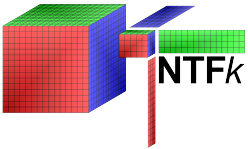
- Feature extraction (**FE**)
- Blind source separation (**BSS**)
- Detection of disruptions / anomalies
- Image recognition
- Separate physics processes
- Discover unknown dependencies and phenomena
- Develop reduced-order/surrogate models
- Identify dependencies between model inputs and outputs
- Guide development of physics models representing the data
- Make predictions
- Optimize data acquisition
- “Label” datasets for supervised ML analyses

Nonnegative Matrix/Tensor Factorization

- Novel LANL-patented, open-source, unsupervised Machine Learning (ML) methods and computational techniques
- Based in matrix/tensor factorization coupled with custom k-means clustering and nonnegativity/sparsity constraints:
  - NMFk: Nonnegative **Matrix** Factorization
  - NTFk: Nonnegative **Tensor** Factorization
  - `https://github.com/TensorDecompositions`
- Capable to efficiently process large datasets (TB's) utilizing GPU's, TPU's & FPGA's

  $\Rightarrow$ julia, Flux.jl, Knet.jl, AutoOffLoad.jl, TensorFlow, PyTorch, MXNet





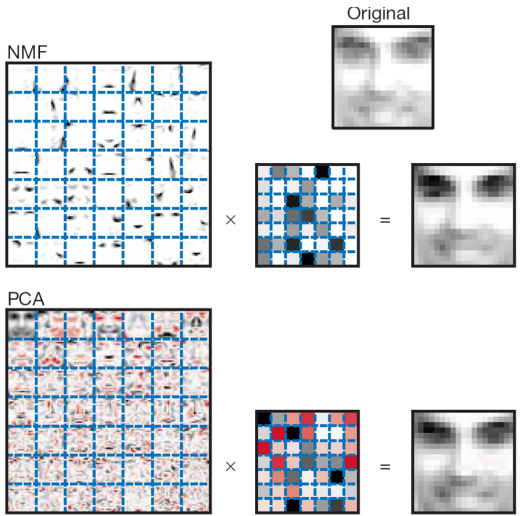
- julia REPL commands:
  - import Pkg
  - Pkg.add("NMFk")
  - Pkg.add("NTFk")
  - Pkg.add("Mads")

Why Julia?

- julia is as fast as C/FORTRAN
- julia is high-level language (easier to use than MATLAB and Python)
- julia allows for low-level coding (anything that can be coded in C)
- julia compiler can be written in julia
- julia is designed for technical computing
- julia packaging system is amazing (module unit testing is required)
- julia code without changes runs on laptop/cluster/cloud and can efficiently use the available resources
- julia is actively developed and its community is bustling
- `https://julialang.org`
- `https://juliacomputing.com`
- `https://discourse.julialang.org`

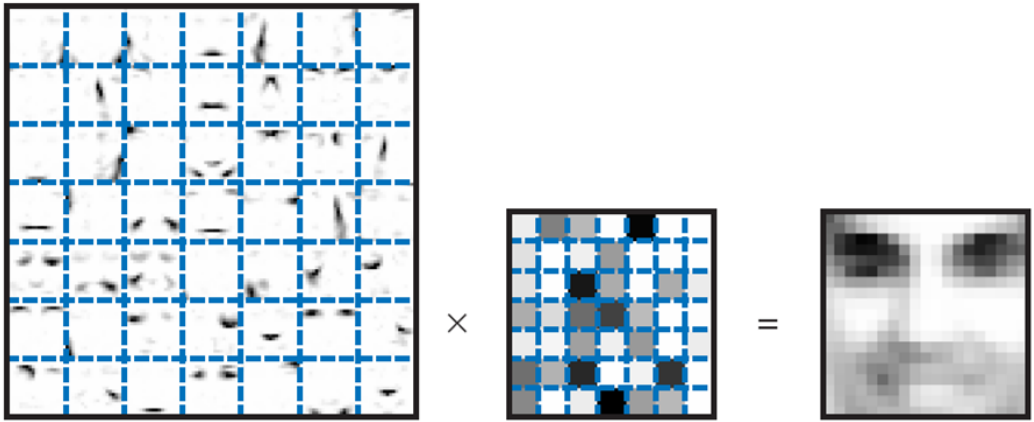
Why nonnegativity?

- NMF vs PCA (Lee & Seung, 1999)
- NMF: Nonnegative Matrix Factorization
- PCA: Principal Component Analysis

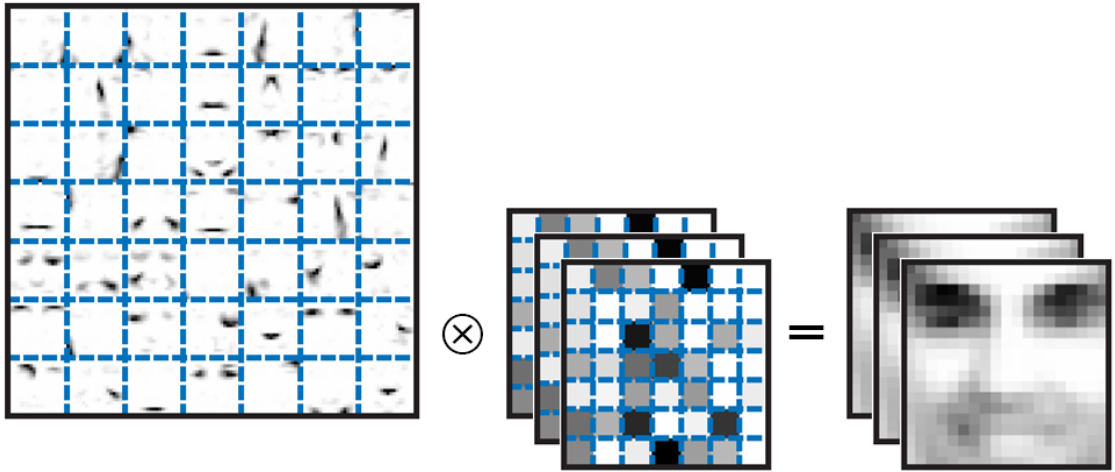


Nonnegativity constraints provide meaningful and interpretable results (+sparsity)

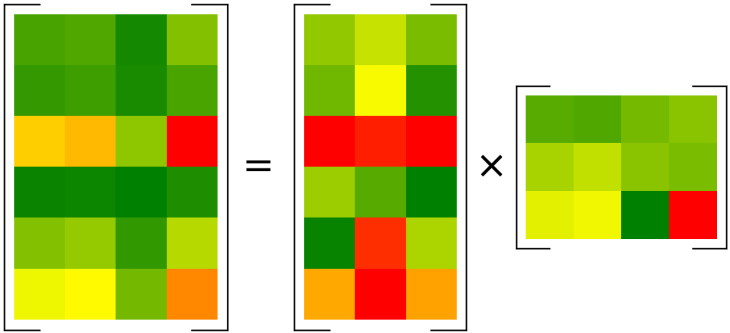
NMF: Nonnegative Matrix Factorization



NTF: Nonnegative Tensor Factorization

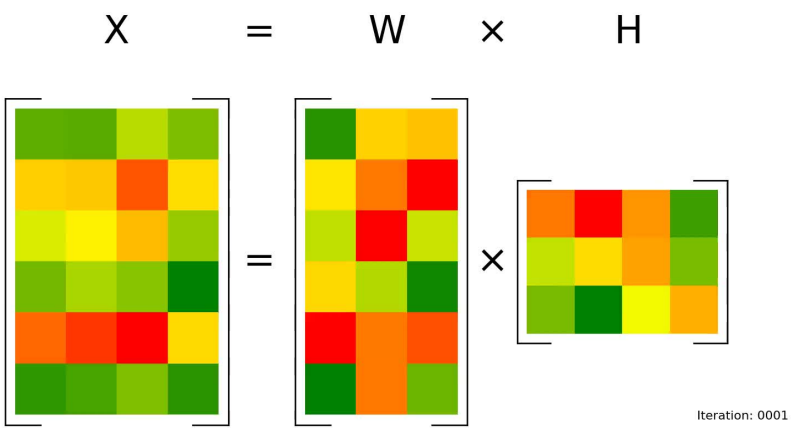


X = W × H

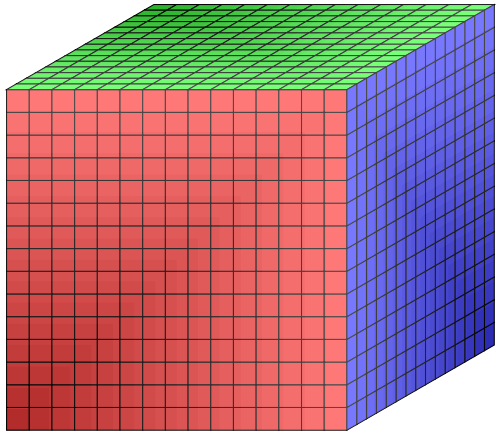


24 knowns $(6 \times 4) \rightarrow 30$ unknowns $(6 \times 3) + (3 \times 4)$
number of features k is also unknown (here, $k = 3$))

NMFk: Factorization process



Tensor Decomposition (3D case): Rank-1 tensor



X
$[K, M, N]$

$\approx$

λ

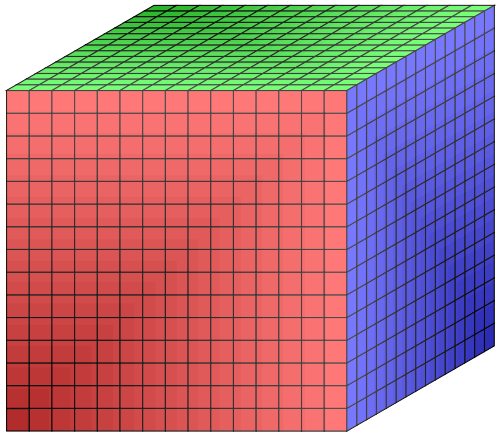
H
$[1, K]$

W
$[1, N]$

V
$[1, M]$

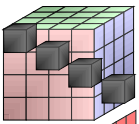
$$X \approx \lambda(H \otimes W \otimes V)$$

Tensor Decomposition (3D case): Rank-4 tensor

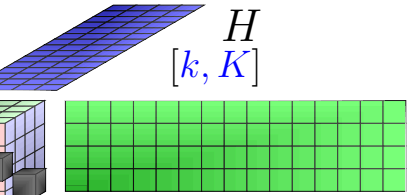


$$X \\ [K, M, N]$$

$\approx$



$$D \\ [k, k, k]$$



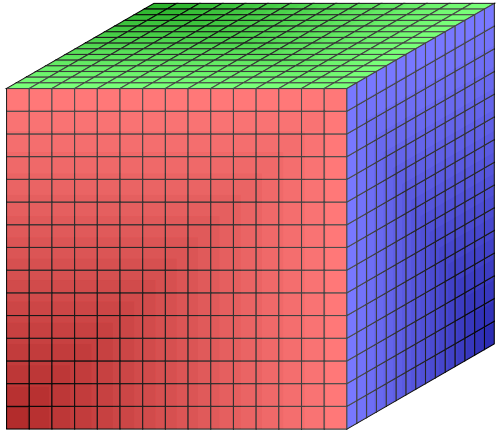
$$H \\ [k, K]$$

$$W \\ [k, N]$$

$$V \\ [k, M]$$

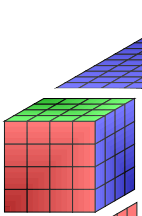
$$X \approx D \otimes H \otimes W \otimes V$$

Tensor Decomposition (3D case): Rank-64 / Multirank-(4,4,4) tensor



X
$[K, M, N]$

$\approx$



G
$[k, k, k]$

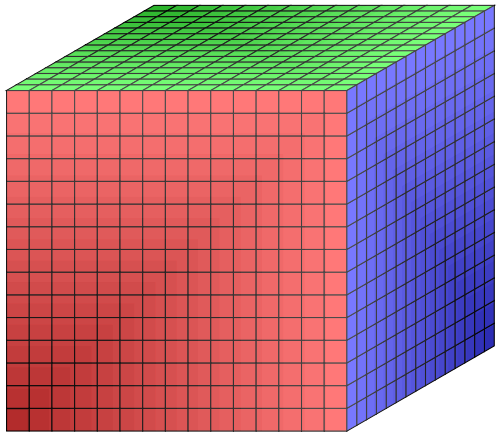
H
$[k, K]$

W
$[k, N]$

V
$[k, M]$

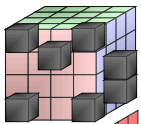
$$X \approx G \otimes H \otimes W \otimes V$$

Tucker Tensor Decomposition (3D case): Rank-7 Multirank-(3,3,4)

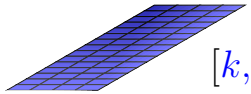


X
$[K, M, N]$

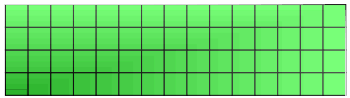
$\approx$



D
$[k, k, k]$



H
$[k, K]$



W
$[k, N]$

V
$[k, M]$

$$X \approx D \otimes H \otimes W \otimes V$$

- **Field Data**:
  - Contamination
  - Climate
  - Geothermal
  - Seismic
  - Oil/gas production
- **Lab Data**:
  - X-ray Spectroscopy
  - UV Fluorescence Spectroscopy
  - Microbial population analyses
  - Isotope fractionation
- **Operational Data**:
  - LANSCE: Los Alamos Neutron Accelerator
  - Oil/gas production
- **Model Outputs**:
  - Reactive mixing $A + B \rightarrow C$
  - Phase separation of co-polymers
  - Molecular Dynamics of proteins
  - Climate modeling

LANL/EES: Projects / Collaborations

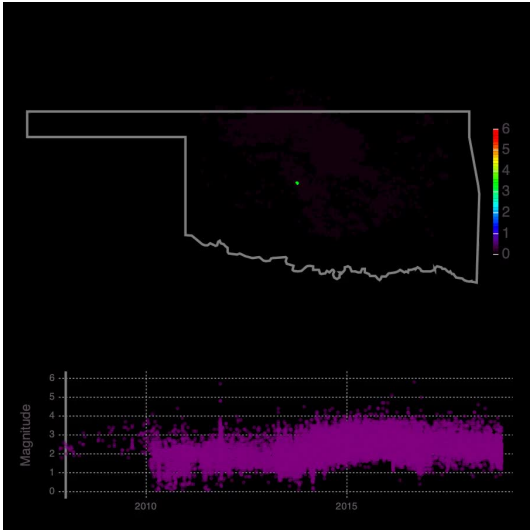
- DOE Office of Science
  AI / ML Grand Challenges (10-15 years out)
- DOE Fossil Energy
- DOE Geothermal
- DOE Carbon Sequestration
- DOE Biological and Environmental Research
- DOE ARPA E
- Industrial partners (Google, JuliaComputing, Descartes Lab, ...)
- Academia (MIT, Stanford, UC, UT, ...)

- Students
- Postdocs
- Scientists

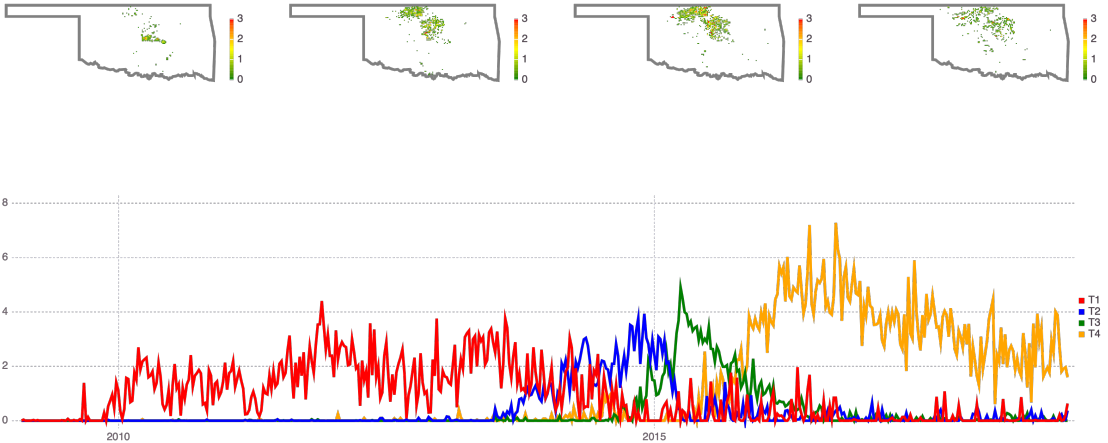
- Vesselinov, Munuduru, Karra, O'Maley, Alexandrov, Unsupervised Machine Learning Based on Non-Negative Tensor Factorization for Analyzing Reactive-Mixing, **Journal of Computational Physics**, Special issue: Machine Learning, 2019.
- Stanev, Vesselinov, Kusne, Antoszewski, Takeuchi, Alexandrov, Unsupervised Phase Mapping of X-ray Diffraction Data by Nonnegative Matrix Factorization Integrated with Custom Clustering, **Nature Computational Materials**, 2018.
- Vesselinov, O'Malley, Alexandrov, Nonnegative Tensor Factorization for Contaminant Source Identification, **Journal of Contaminant Hydrology**, 2018.
- O'Malley, Vesselinov, Alexandrov, Alexandrov, Nonnegative/binary matrix factorization with a D-Wave quantum annealer, **PLOS ONE**, 2018.
- Vesselinov, O'Malley, Alexandrov, Contaminant source identification using semi-supervised machine learning, **Journal of Contaminant Hydrology**, 2017.
- Alexandrov, Vesselinov, Blind source separation for groundwater level analysis based on nonnegative matrix factorization, **WRR**, 2014.

Oklahoma seismicity

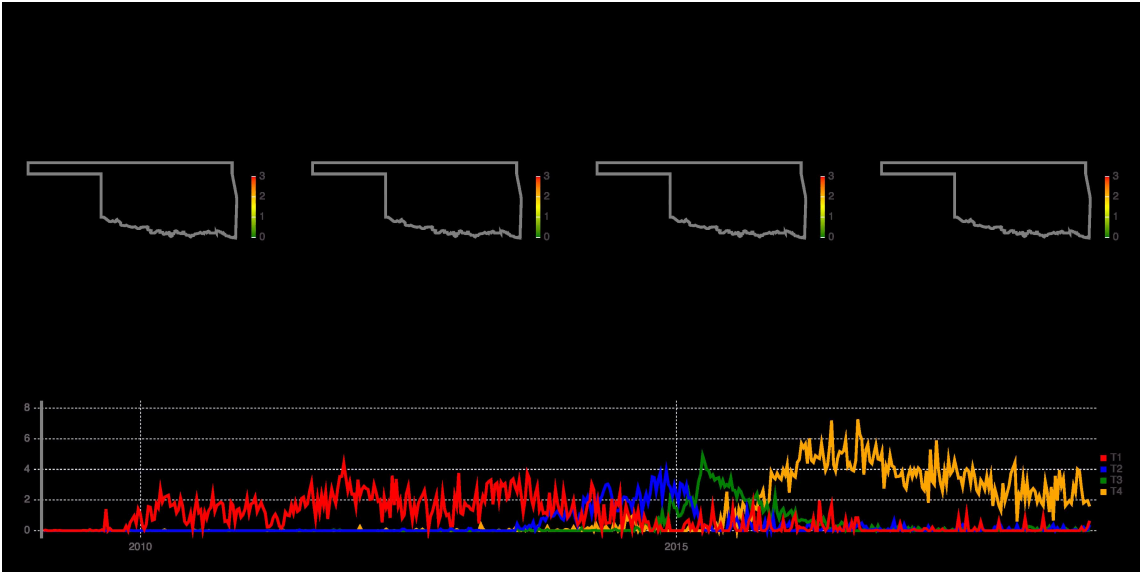
- 32,251 seismic events from 1989 to 2017
- Tensor: total energy of events over a discretized domain
- **NTF**k extracts spatial footprints and temporal patterns of dominant hidden (latent) features



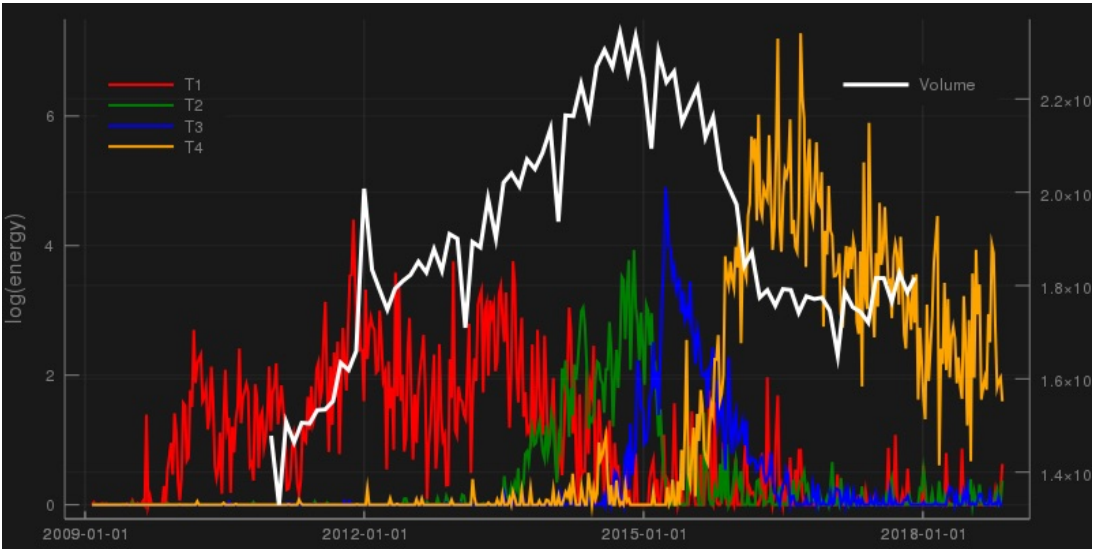
Oklahoma seismicity: reconstruction by 4 features (signals)



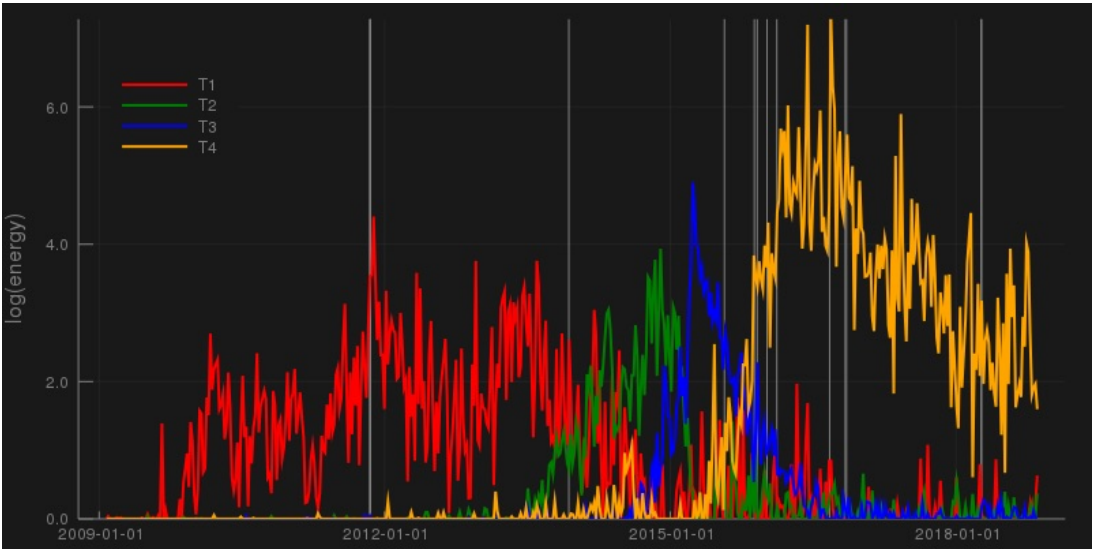
Oklahoma seismicity: reconstruction by 4 features (signals)



Oklahoma seismicity: extracted signals vs. injected volumes

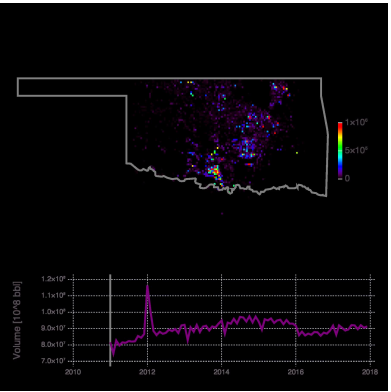


Oklahoma seismicity: extracted signals vs. majors seismic events

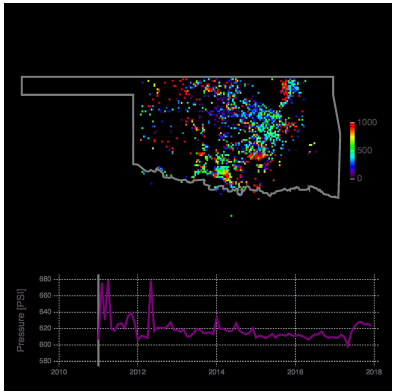


Oklahoma seismicity

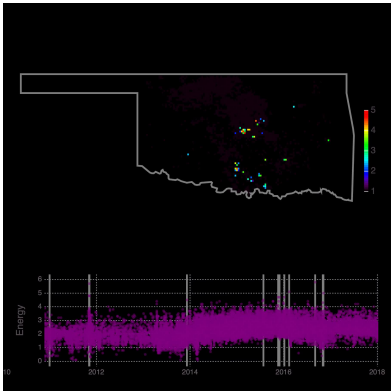
volume



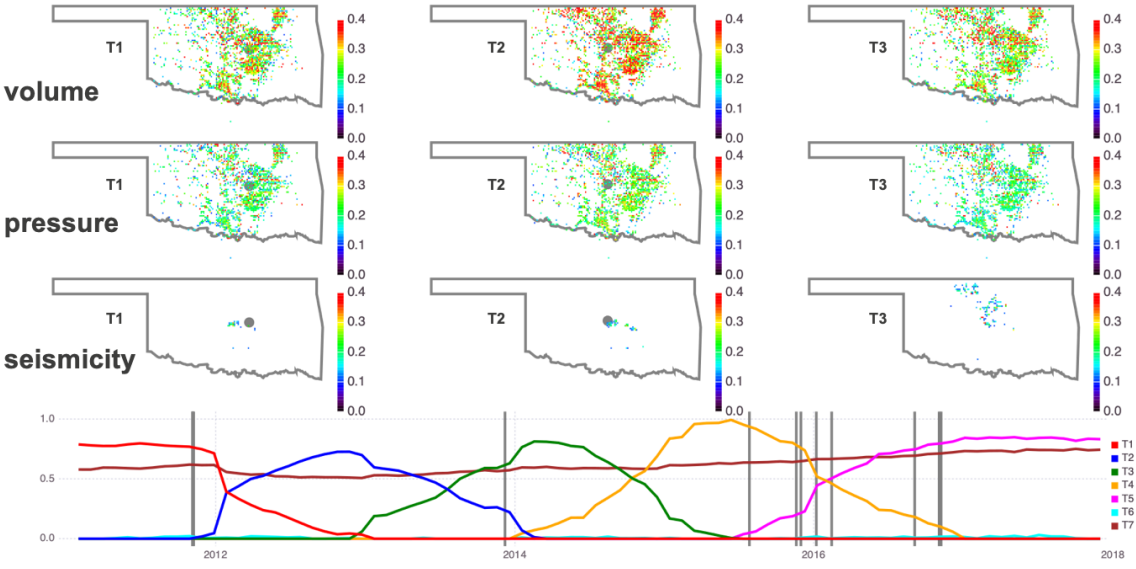
pressure



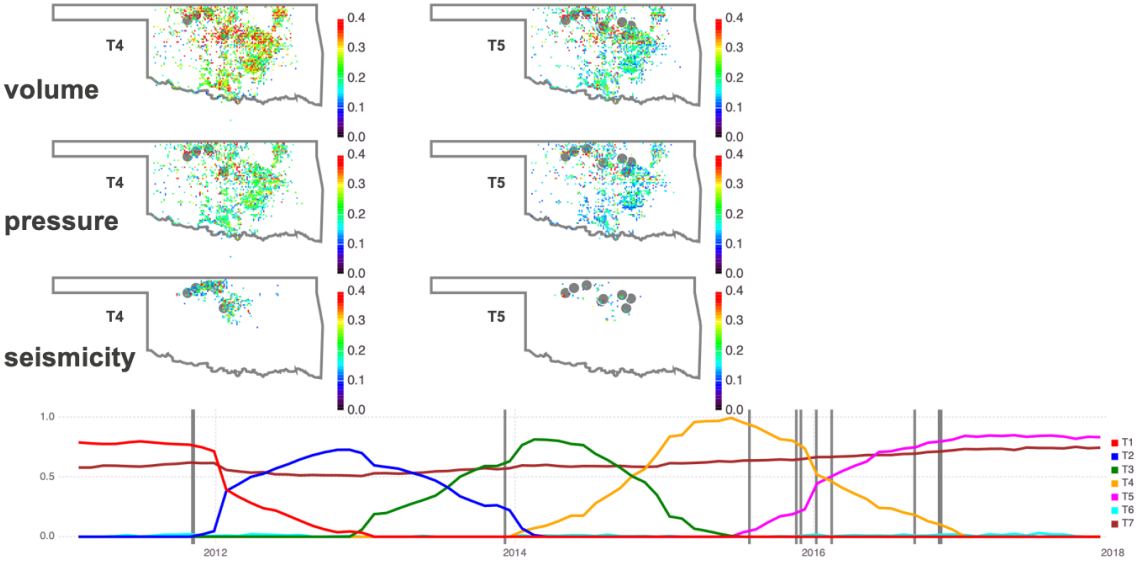
seismicity



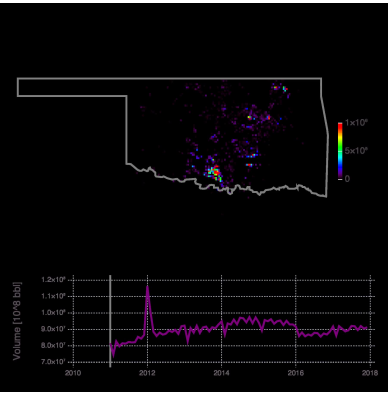
Oklahoma seismicity: 5 volume/pressure/seismicity features



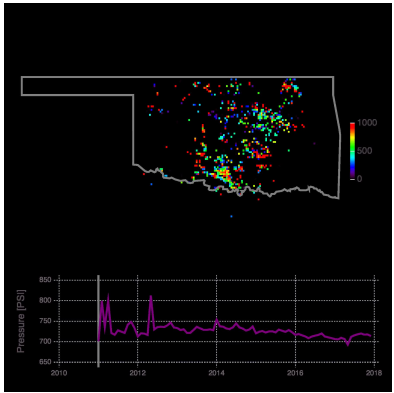
Oklahoma seismicity: 5 volume/pressure/seismicity features



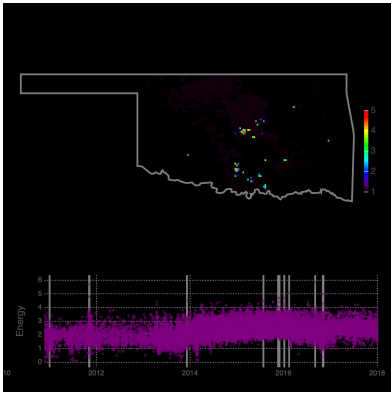
volume recovery



pressure recovery



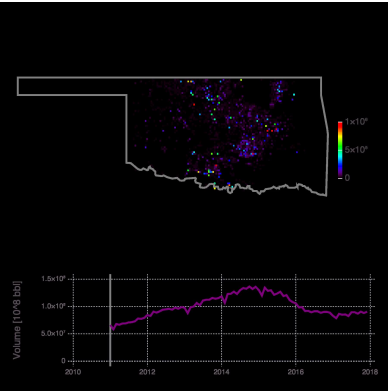
seismicity



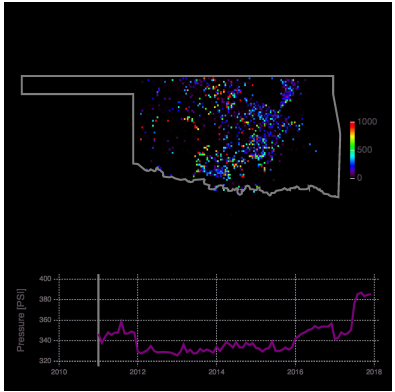
Recovery injection has limited impact on seismicity

Oklahoma seismicity: Disposal injection

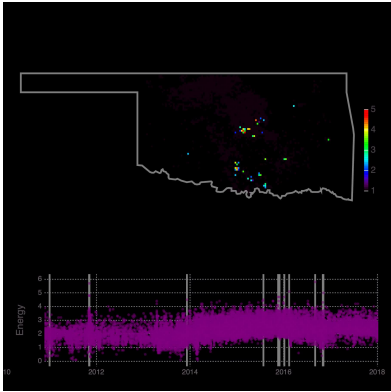
volume disposal



pressure disposal

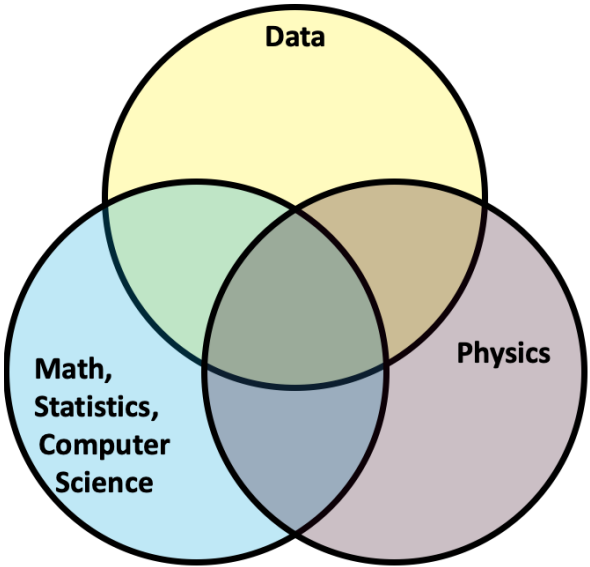


seismicity

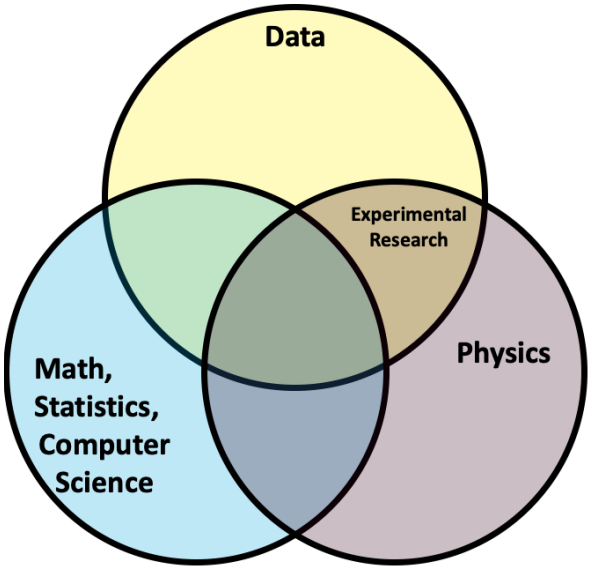


Disposal injection has impact on seismicity

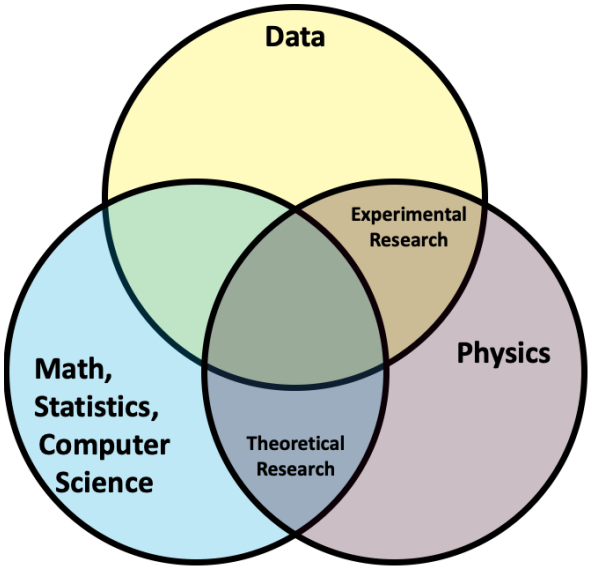
Physics-Informed Machine Learning



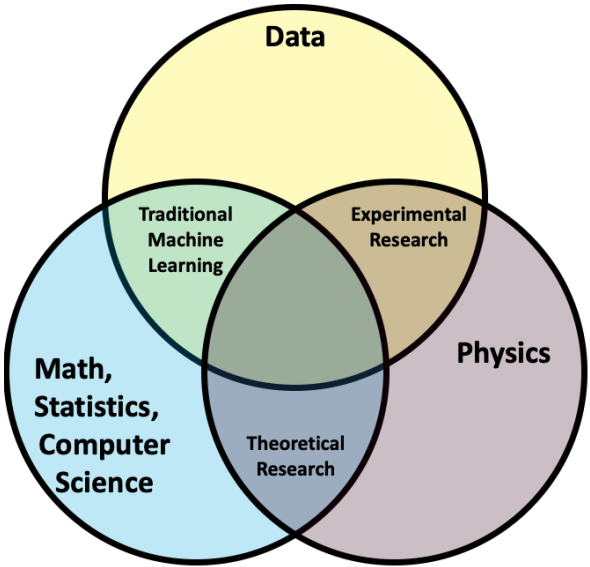
Physics-Informed Machine Learning



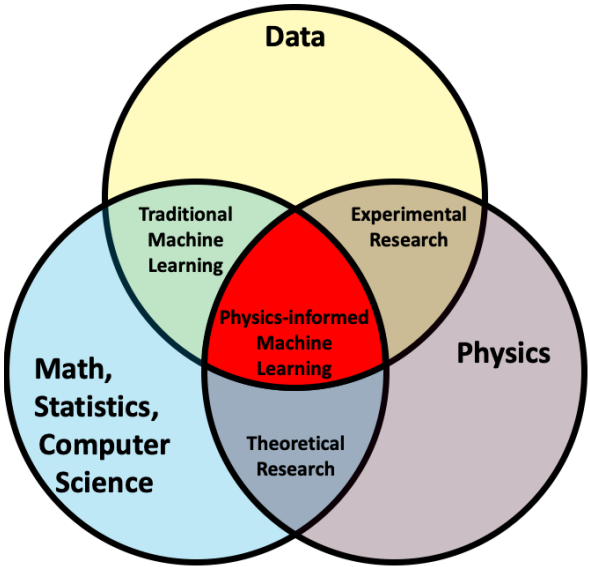
Physics-Informed Machine Learning



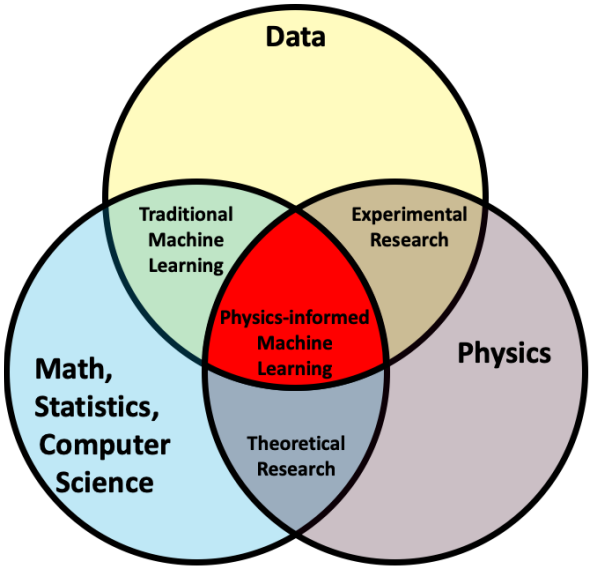
Physics-Informed Machine Learning



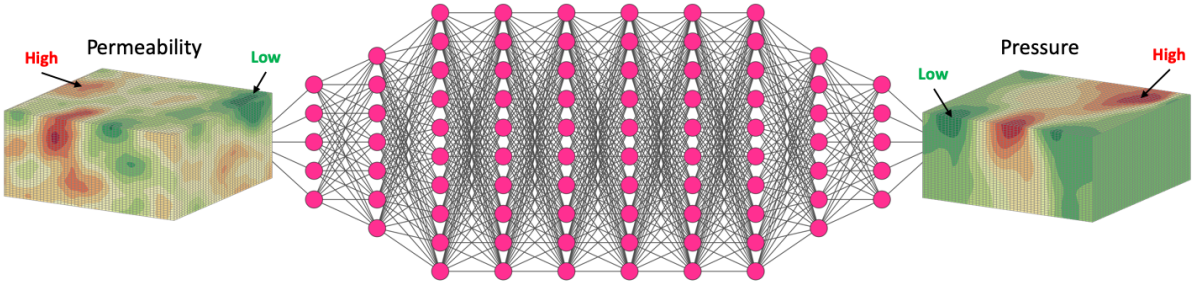
Physics-Informed Machine Learning



- **Empirical**: observations and experiments (since the cradle of our civilization)
- **Theoretical**: generalizations and models (since 1600's)
- **Computational**: analytical and numerical simulations (since 1950's)
- **Data-exploration**: unify data, simulations, and theory (since 2000's)

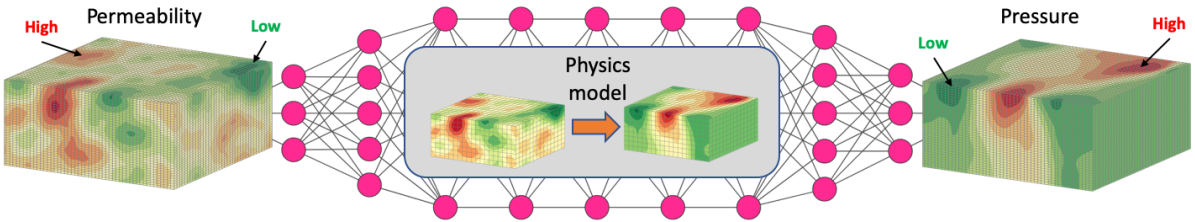


Traditional Machine Learning Neural Networks



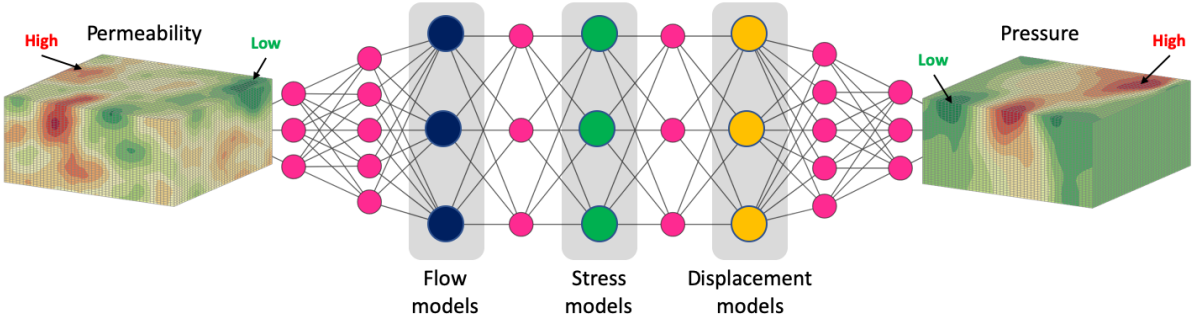
- it is a **black box**, **ad hoc** approach
- no preconceived knowledge about analyzed problem (general)
- all the neurons are $relu(Ax + b)$; A and b have no physical meaning; $relu()$ does not impose physics constraints
- neural networks needs to be very **deep** and **wide** to represent complex physics

Physics-Informed Machine Learning Neural Networks



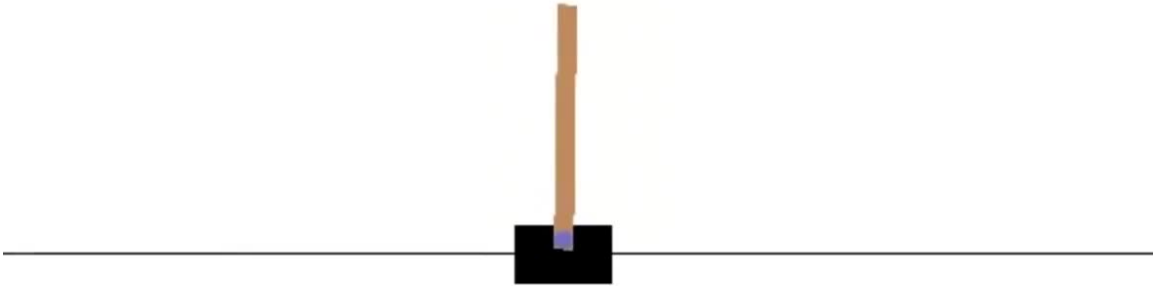
- include preconceived knowledge about analyzed problem (problem specific)
- neurons can represent $PhysicsModel(Ax + b)$; A and b have physical interpretation; $PhysicsModel()$ imposes physics constraints (e.g. conservation of mass/species)
- **PIML** models can be **trained (optimized) faster** and with **less training data**

Physics-Informed Machine Learning Neural Networks

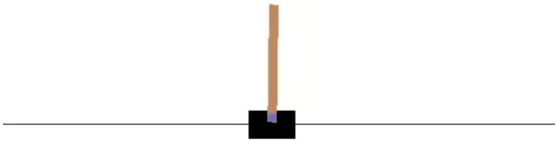


- physics-informed layers (**"fat" neurons**) capture important governing processes (e.g., flow, stress, deformation, and displacement)
- can be done only through differentiable programming in julia

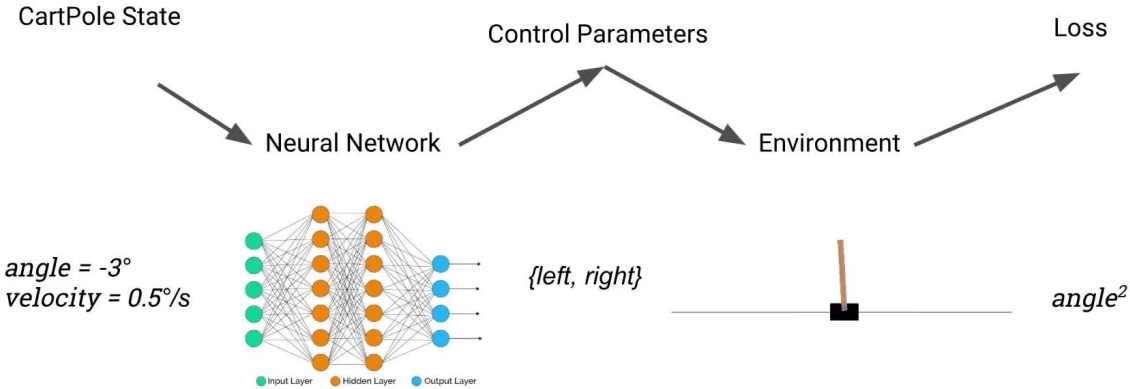
Physics-Informed Machine Learning (PIML): Cartpole



Physics-Informed Machine Learning (PIML): Cartpole

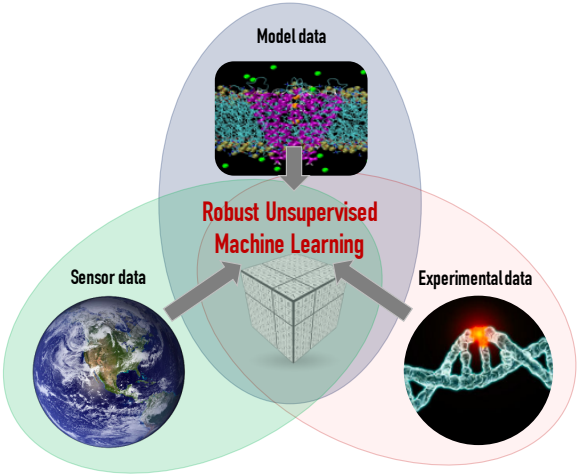


Physics-Informed Machine Learning (PIML): Cartpole



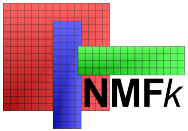
Summary

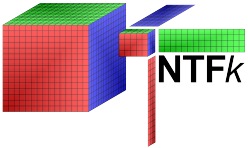
- Developed **novel** unsupervised and physics-informed ML methods and computational tools
- Our ML methods have been used to solve various real-world problems (brought breakthrough discoveries related to human cancer research)
- julia



- Codes:

NMFk MADS NTFk





- Examples:

http://madsjulia.github.io/Mads.jl/Examples/blind_source_separation
http://tensors.lanl.gov
http://tensordecompositions.github.io
https://github.com/TensorDecompositions
https://hub.docker.com/u/montyvesselinov

julia