Machine Learning for Inertial Fusion

Ryan G. McClarren
University of Notre Dame
Dept. of Aerospace and Mechanical Engineering

Acknowledgements

- Much of what you see here comes from Kelli Humbird's dissertation work. She has since gone on to do great work in ICF at Lawrence Livermore Laboratory.
- Though their work does not appear here, I would also like to thank
  - My current graduate students: Q. Lan, B. Whewell, M. Vander Wal, W. Bennett, E. Smith, and S. Pasmann
  - Minwoo Shin and Ilham Variansyah, current postdocs in my group.

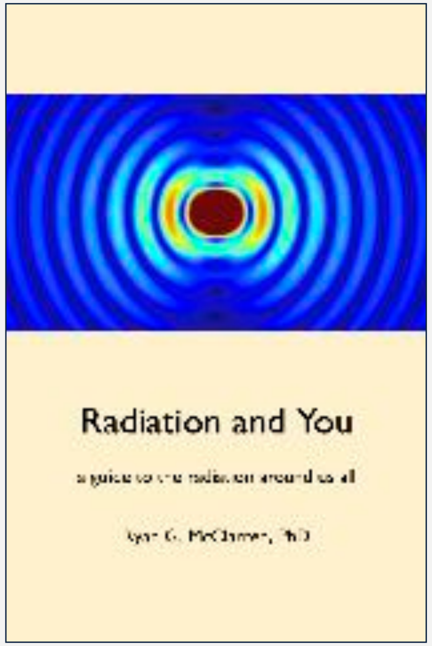



In case you want to hear more from me, I have several books available:

Uncertainty Quantification and Predictive Computational Science from Springer https://www.springer.com/gp/book/9783319995243

Computational Nuclear Engineering and Radiological Science Using Python from Academic Press http://a.co/2HdisVb

Radiation and You is a children's book (ages 7-13) with lots of pictures about how radiation is all around us and how it is used. It is available from Orion Scientific Publishing http://a.co/92FpGeK




19

Radiation for transportation

Nuclear power can also be used to power ships and other types of transportation. In the navy, nuclear reactors power submarines and other ships, such as aircraft carriers. The use of nuclear reactors in ships allows them to go a long time without needing more fuel. Some nuclear submarines do not need to be refueled for 25 years—the fuel lasts longer than the ship itself. Also, because nuclear reactors do not need oxygen to produce energy, these submarines can stay underwater for a long time.

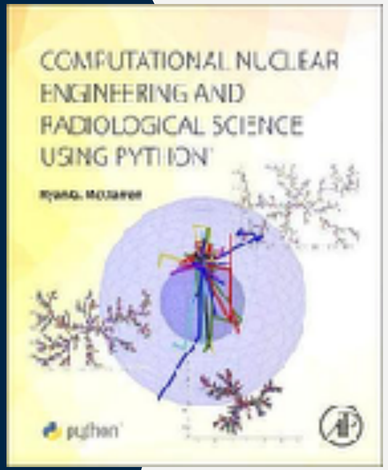
Nuclear energy is also used to power spacecraft. Many of the spacecraft sent out from Earth have radioactive atoms. When these atoms decay, heat is produced, and this heat is used to produce electricity, like a nuclear battery. A common radioactive atom in these nuclear batteries is plutonium-238, which is produced in nuclear reactors. Nuclear batteries can give power for decades. The Voyager probes have been working for almost 40 years on nuclear batteries and still send messages to Earth.

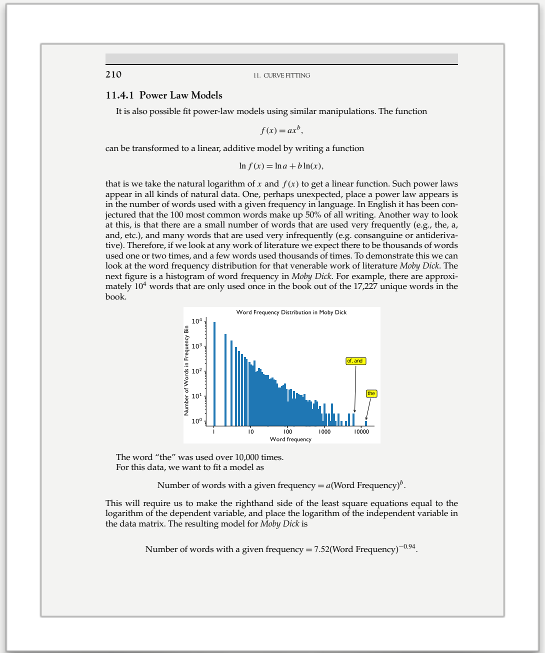
Three nuclear-powered ships, USS Enterprise (bottom), USS Long Beach (center) and USS Bainbridge (top) sailing in the Mediterranean Sea. Sailors on the Enterprise are standing to form Einstein's famous equation: $E = mc^2$.

The New Horizons spacecraft traveled from Earth to Pluto in just over nine years. The spacecraft was powered by a plutonium-238 battery. The mission to Pluto was fueled by plutonium.

← This is a picture of the USS Michigan, a nuclear-powered submarine, while at a dock. This submarine could stay at sea for 90 days.

This plutonium-238 was created for a nuclear battery. It is glowing because the radiation it produces makes it hot.




210 11. CURVE FITTING

11.4.1 Power Law Models

It is also possible fit power-law models using similar manipulations. The function

$$f(x) = ax^b,$$

can be transformed to a linear, additive model by writing a function

$$\ln f(x) = \ln a + b\ln(x),$$

that is we take the natural logarithm of x and $f(x)$ to get a linear function. Such power laws appear in all kinds of natural data. One, perhaps unexpected, place a power law appears is in the number of words used with a given frequency in language. In English it has been conjectured that the 100 most common words make up 50% of all writing. Another way to look at this, is that there are a small number of words that are used very frequently (e.g., the, a, and, etc.), and many words that are used very infrequently (e.g. consanguine or antiderivative). Therefore, if we look at any work of literature we expect there to be thousands of words used one or two times, and a few words used thousands of times. To demonstrate this we can look at the word frequency distribution for that venerable work of literature *Moby Dick*. The next figure is a histogram of word frequency in *Moby Dick*. For example, there are approximately 10^4 words that are only used once in the book out of the 17,227 unique words in the book.

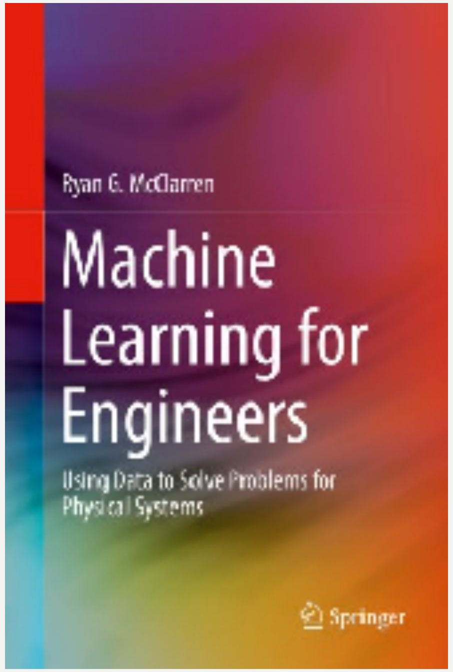
Word Frequency Distribution in Moby Dick

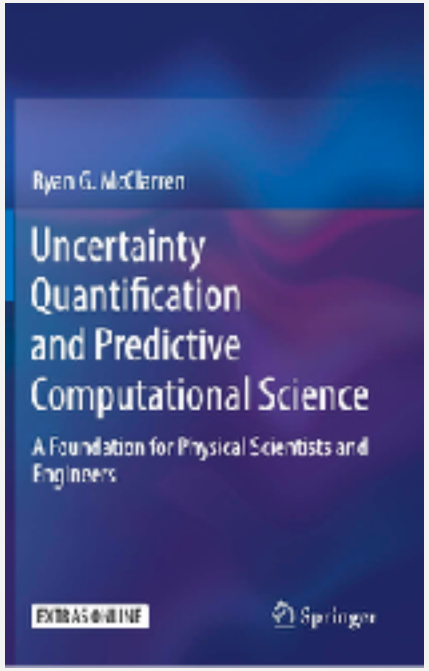
The word "the" was used over 10,000 times.
For this data, we want to fit a model as

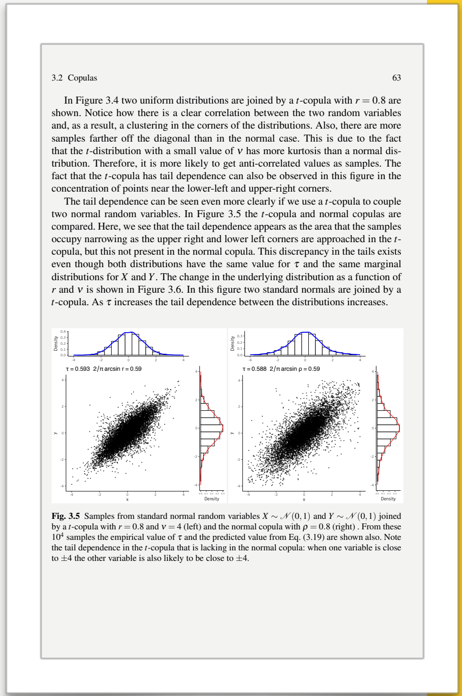
$$\text{Number of words with a given frequency} = a(\text{Word Frequency})^b.$$

This will require us to make the righthand side of the least square equations equal to the logarithm of the dependent variable, and place the logarithm of the independent variable in the data matrix. The resulting model for *Moby Dick* is

$$\text{Number of words with a given frequency} = 7.52(\text{Word Frequency})^{-0.94}.$$






3.2 Copulas 63

In Figure 3.4 two uniform distributions are joined by a t-copula with $r = 0.8$ are shown. Notice how there is a clear correlation between the two random variables and, as a result, a clustering in the corners of the distributions. Also, there are more samples farther off the diagonal than in the normal case. This is due to the fact that the t-distribution with a small value of ν has more kurtosis than a normal distribution. Therefore, it is more likely to get anti-correlated values as samples. The fact that the t-copula has tail dependence can also be observed in this figure in the concentration of points near the lower-left and upper-right corners.

The tail dependence can be seen even more clearly if we use a t-copula to couple two normal random variables. In Figure 3.5 the t-copula and normal copulas are compared. Here, we see that the tail dependence appears as the area that the samples occupy narrowing as the upper right and lower left corners are approached in the t-copula, but this not present in the normal copula. This discrepancy in the tails exists even though both distributions have the same value for τ and the same marginal distributions for X and Y. The change in the underlying distribution as a function of r and ν is shown in Figure 3.6. In this figure two standard normals are joined by a t-copula. As τ increases the tail dependence between the distributions increases.

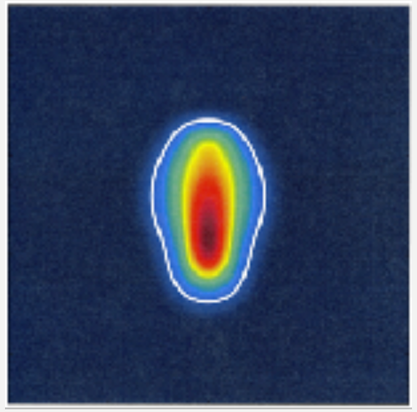
$\tau = 0.593$ $2/\pi \arcsin r = 0.59$ $\tau = 0.588$ $2/\pi \arcsin \rho = 0.59$

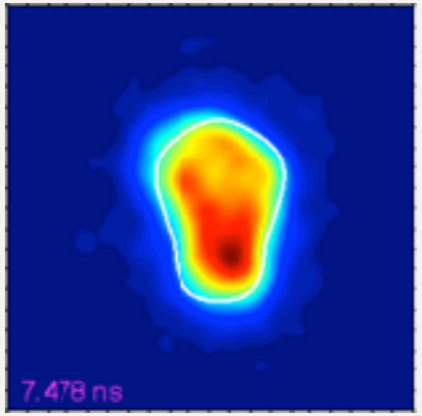
Fig. 3.5 Samples from standard normal random variables $X \sim \mathcal{N}(0,1)$ and $Y \sim \mathcal{N}(0,1)$ joined by a t-copula with $r = 0.8$ and $\nu = 4$ (left) and the normal copula with $\rho = 0.8$ (right) . From these 10^4 samples the empirical value of τ and the predicted value from Eq. (3.19) are shown also. Note the tail dependence in the t-copula that is lacking in the normal copula: when one variable is close to ± 4 the other variable is also likely to be close to ± 4.

New data analysis tools can improve how we design and understand ICF implosions

Optimize design
with simulations

ICF Experiment





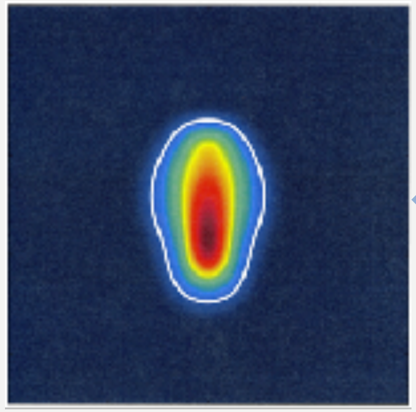
Re-optimize in light of experimental evidence

New data analysis tools can improve how we design and understand ICF implosions

Optimize design with simulations

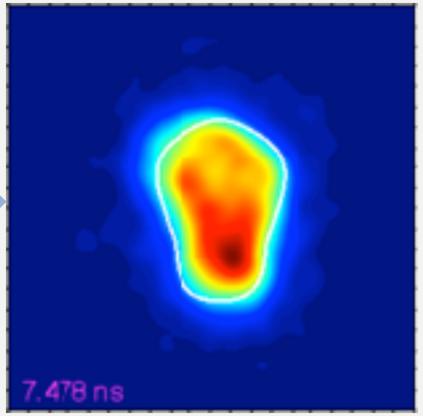
ICF Experiment

How can we better explore vast design spaces for "optimal" implosions?



Are these consistent?

Are there other explanations for the data?

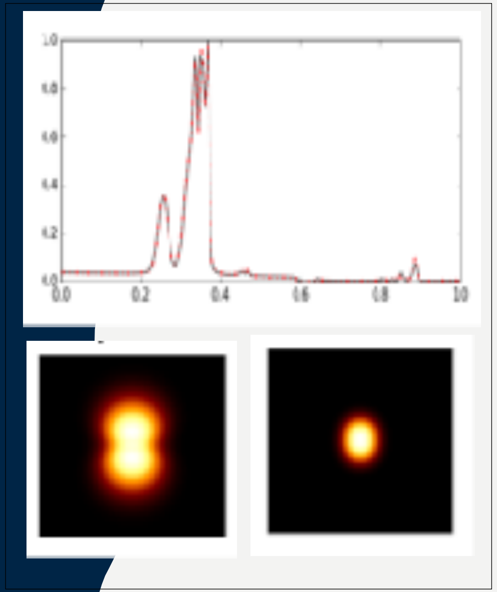


How do we use experimental data to update our models?

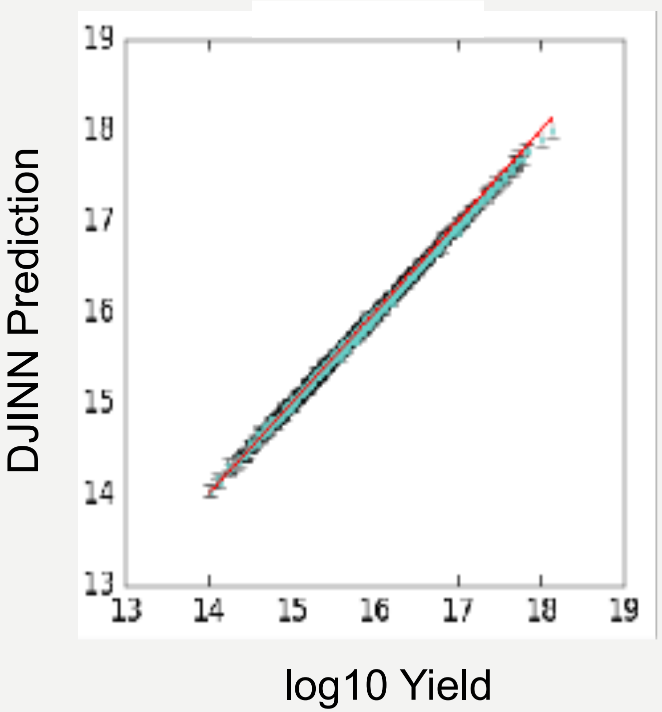
Re-optimize in light of experimental evidence

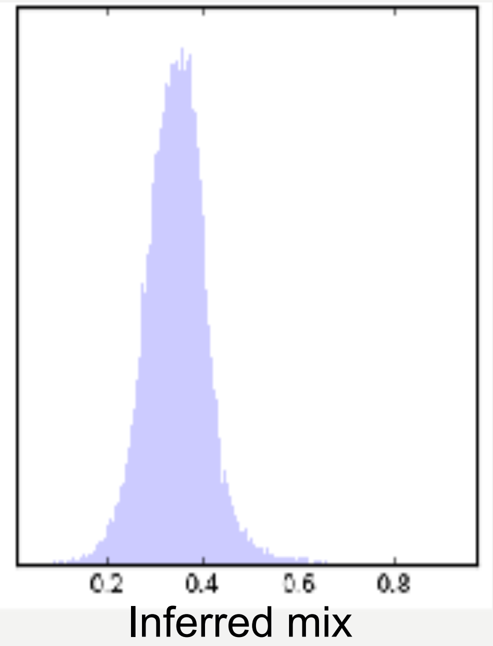
We are using machine learning to integrate simulations and experiments into a common, predictive framework

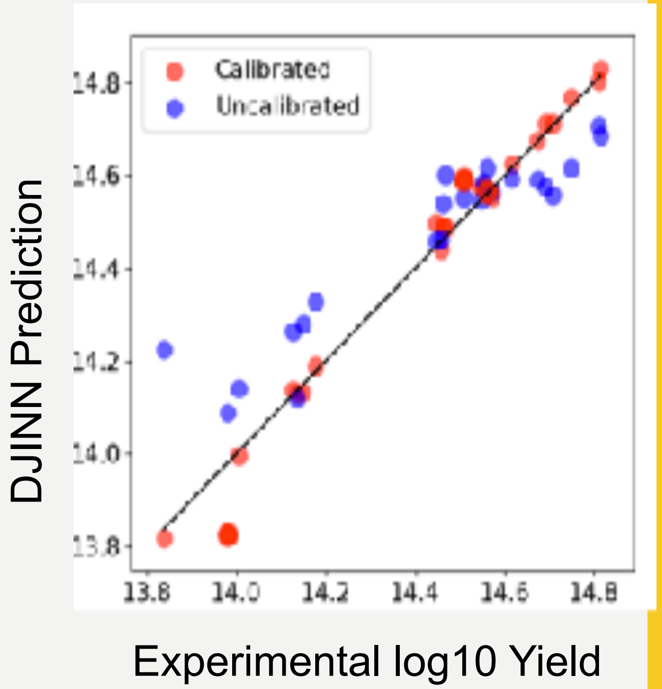
Generate simulation database	Train ML models to emulate ICF codes	Use models to infer unknown inputs	Calibrate simulation models to experiments



Synthetic Diagnostics





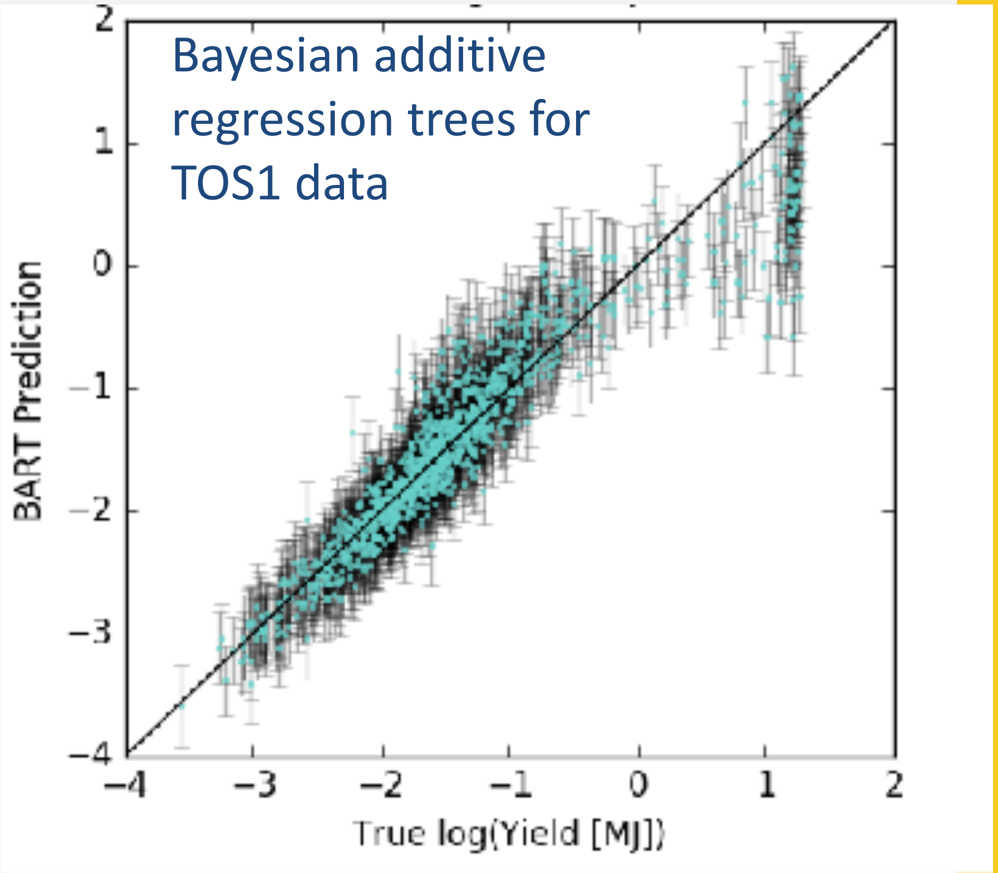


Integrating simulations and experiments:

- Augmenting post-shot analysis with deep neural networks
- Transfer learning to predict Omega ICF experiments

We need prediction uncertainty estimates for quantitative comparison between simulations and experiments

- Uncertainties estimates are needed for quantitative comparison between simulations and experiments
- Traditional Bayesian surrogate models have several drawbacks

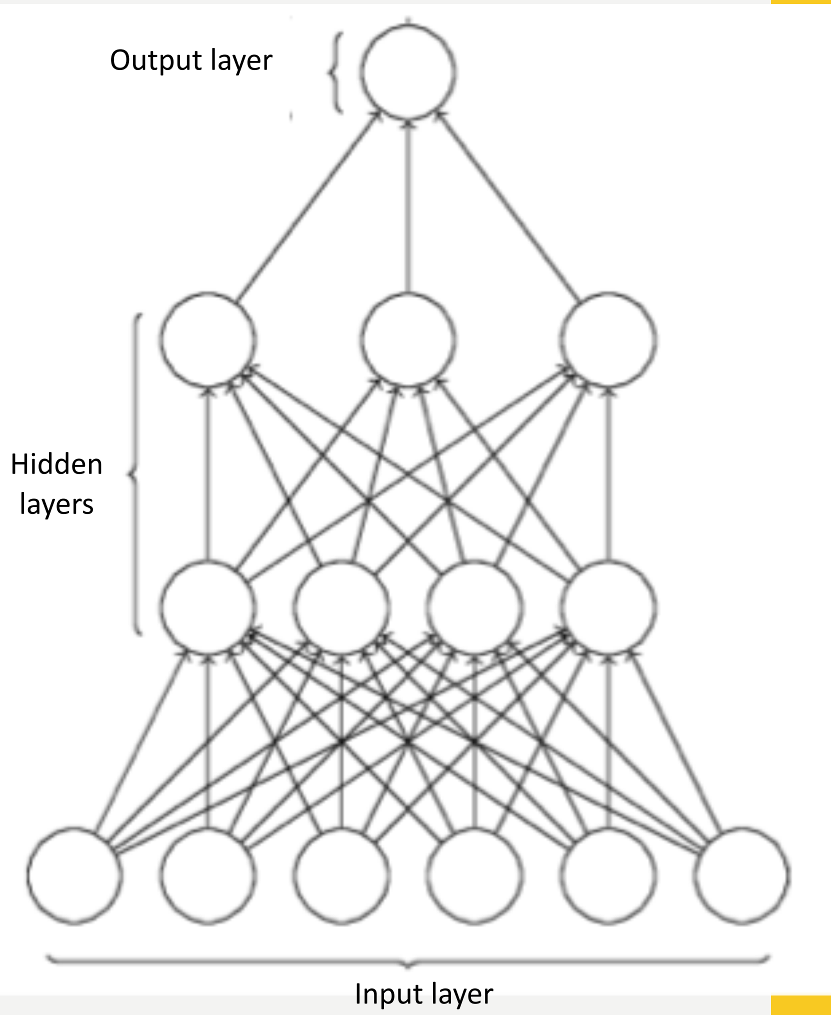


240 MB to store model, ~45 min training time

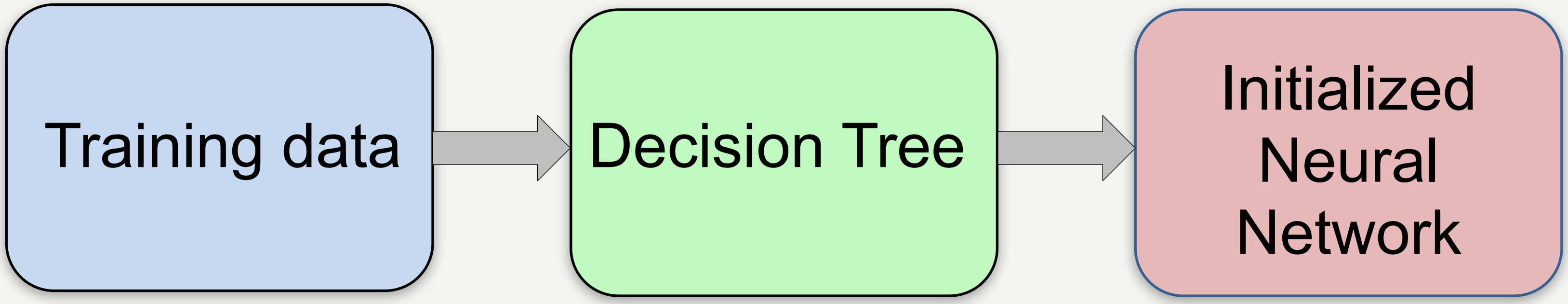
We need fast and scalable surrogates with uncertainties that accurately fit ICF data

Neural networks are notoriously challenging to train

- Model performance is highly dependent on many user-specified hyper-parameters
- No robust guidelines for how to choose hyper-parameters -> hand tuned by experts for specific datasets
- State-of-the-art network design algorithms perform exhaustive searches for optimal settings



Deep Jointly-Informed Neural Networks (DJINN) use decision trees to automatically design and initialize neural networks

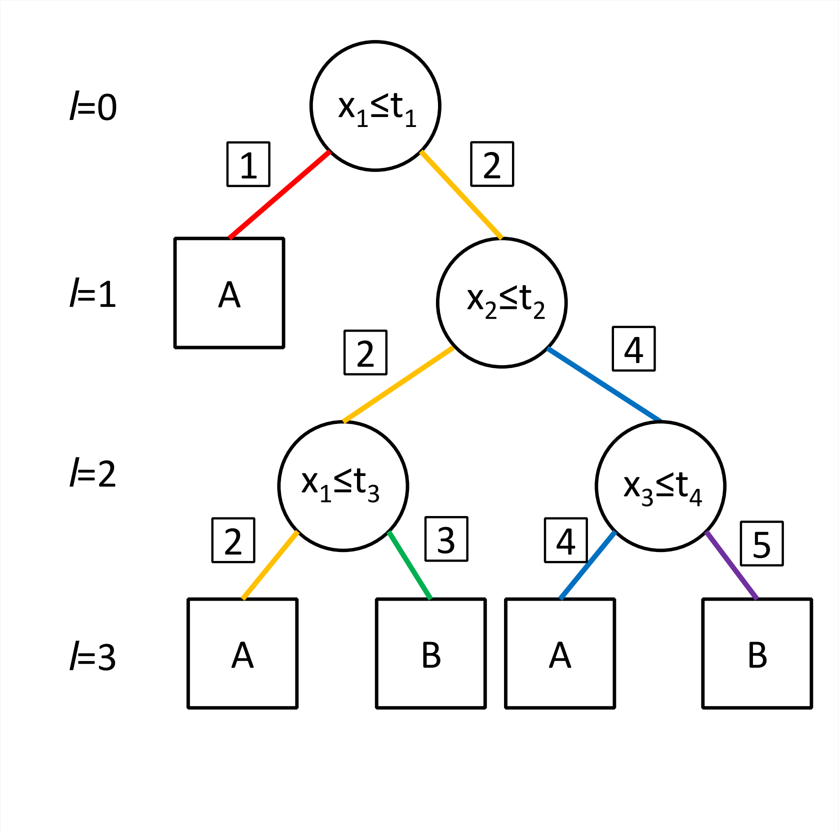


DJINN combines the ease of use of decision trees with the accuracy and scalability of deep neural networks

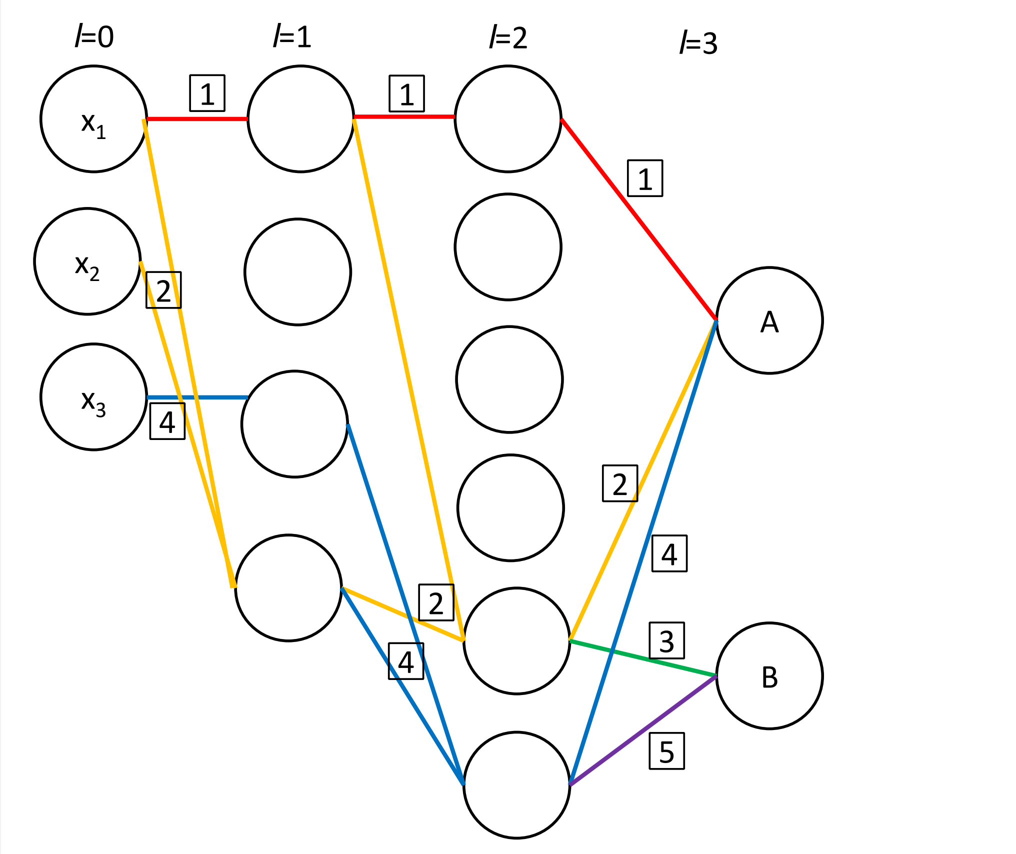
"Deep Neural Network Initialization With Decision Trees"
Kelli D. Humbird ; J. Luc Peterson ; Ryan G. McClarren, IEEE TNNLS (2018).
Accepted, early access: 10.1109/TNNLS.2018.2869694

DJINN maps decision trees to initialized deep feed-forward neural networks

Decision tree



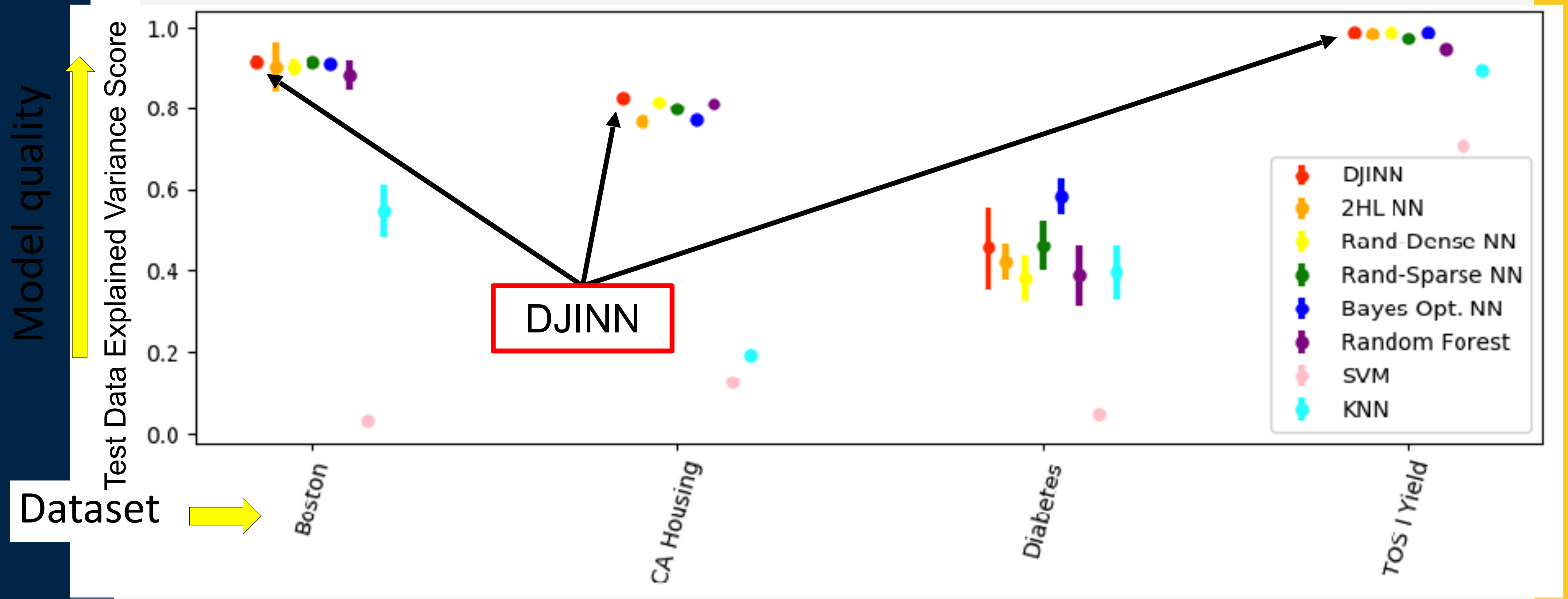
Initialized neural network



DJINN mapping uses tree structure to set neural network architecture, and initializes weights to reflect decision paths through the tree

DJINN often outperforms many other black-box machine learning algorithms and neural network design techniques

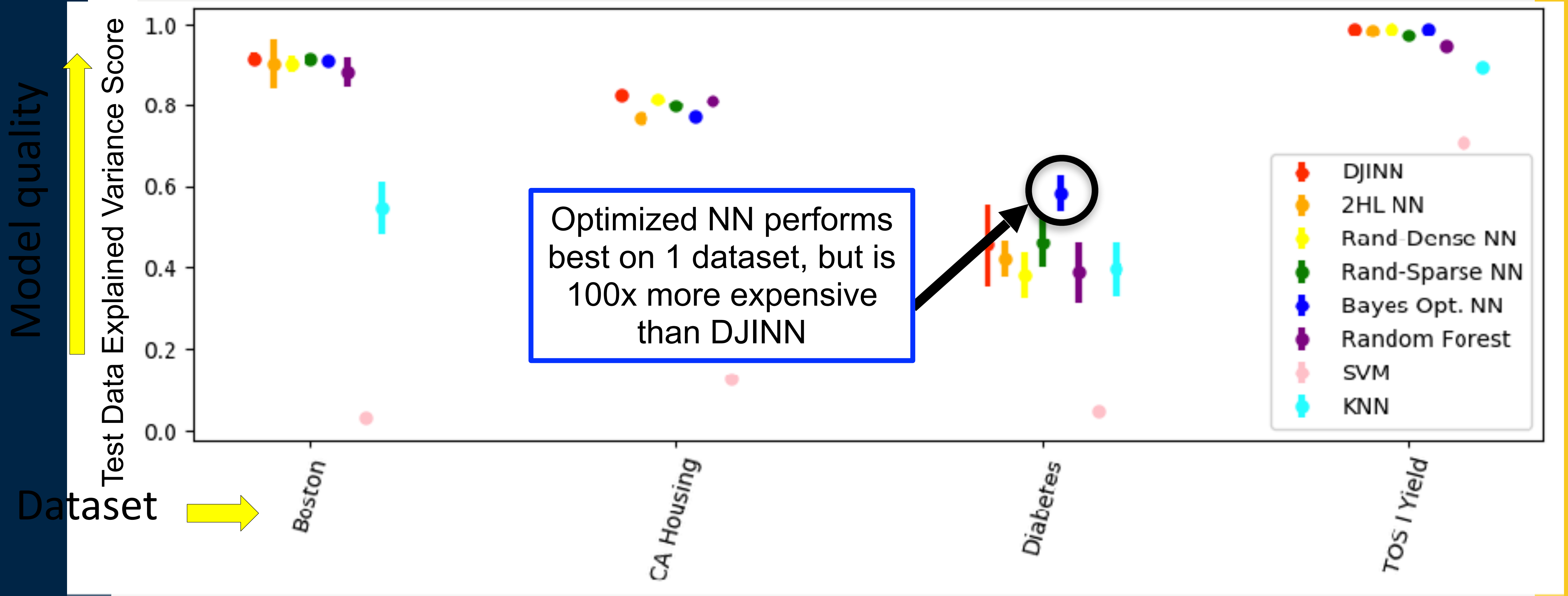
*Error bars display variance in 5x cross-validation scores



* "Neural Random Forests", G. Biau et al, arXiv:1604.07143, 2016.

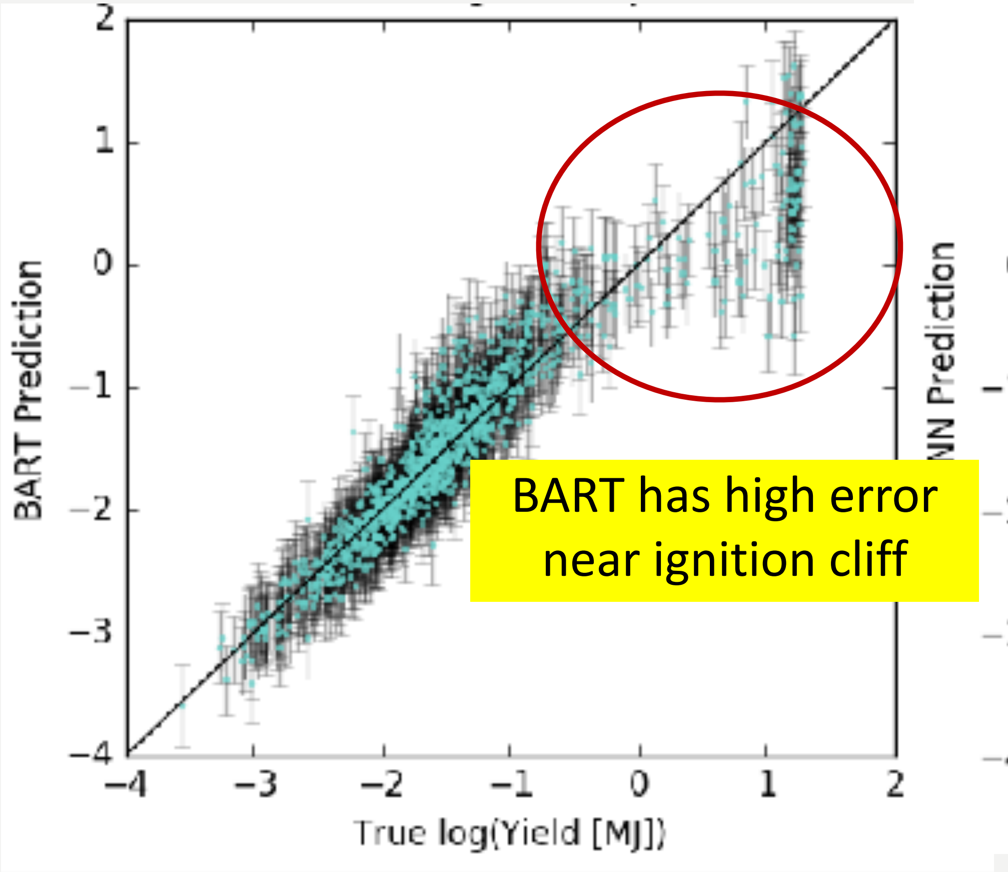
DJINN often outperforms many other black-box machine learning algorithms and neural network design techniques

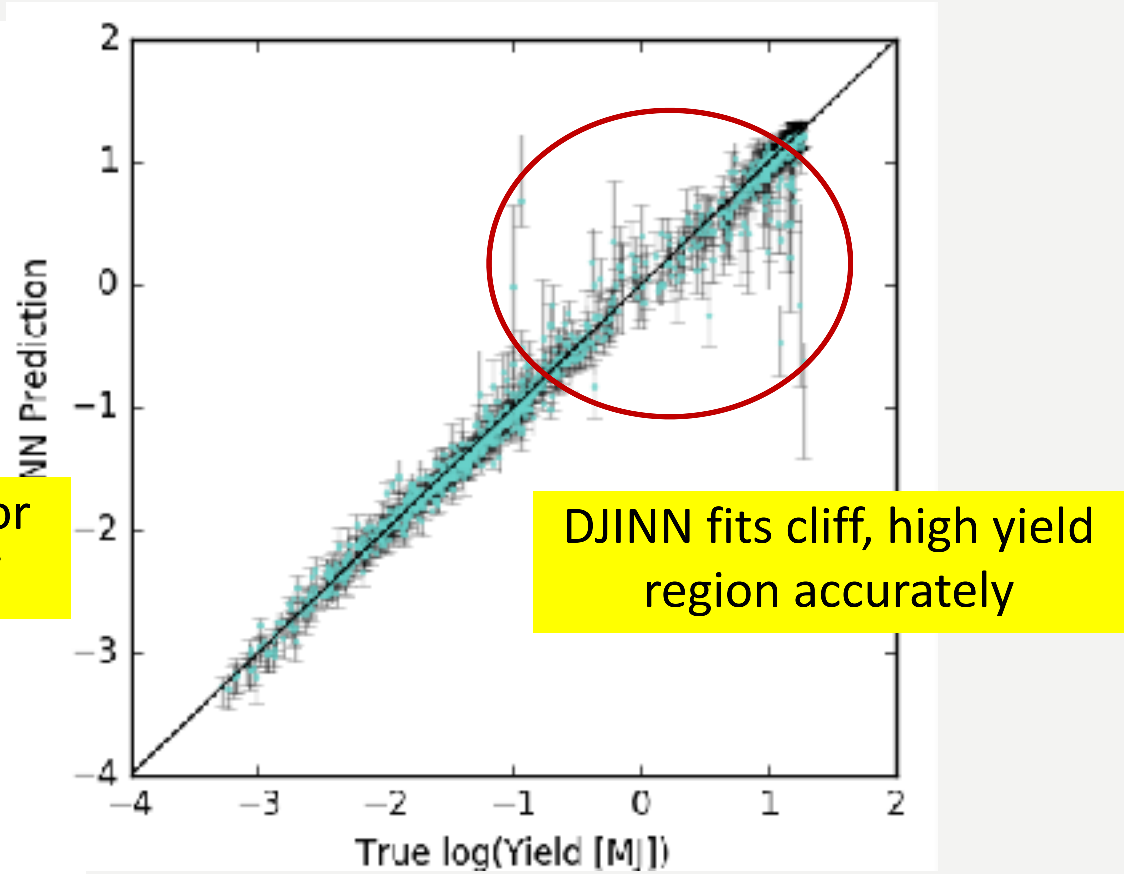
*Error bars display variance in 5x cross-validation scores



* "Neural Random Forests", G. Biau et al, arXiv:1604.07143, 2016.

DJINN with dropout produces models that provide uncertainties on predictions





240 MB to store model,
~45 min training time

900 KB to store model,
~5 min training time

Gal and Ghahramani, "Dropout as a Bayesian Approximation", 2016.

DJINN enables efficient parameter inference and model calibration tasks for merging simulation and experimental data

Parameter Inference

- Inferring unknown simulation parameters using experimental measurements
- Often assumes simulator has low error, but inputs are not well known
- Commonly used to understand the data after the experiment (post-shot analysis)

Model Calibration

- Use experimental data to adjust simulation predictions
- Often assumes simulation inputs are known, but simulator has error
- Commonly used to predict outcome of future experiments

DJINN enables efficient parameter inference and model calibration tasks for merging simulation and

Parameter Inference

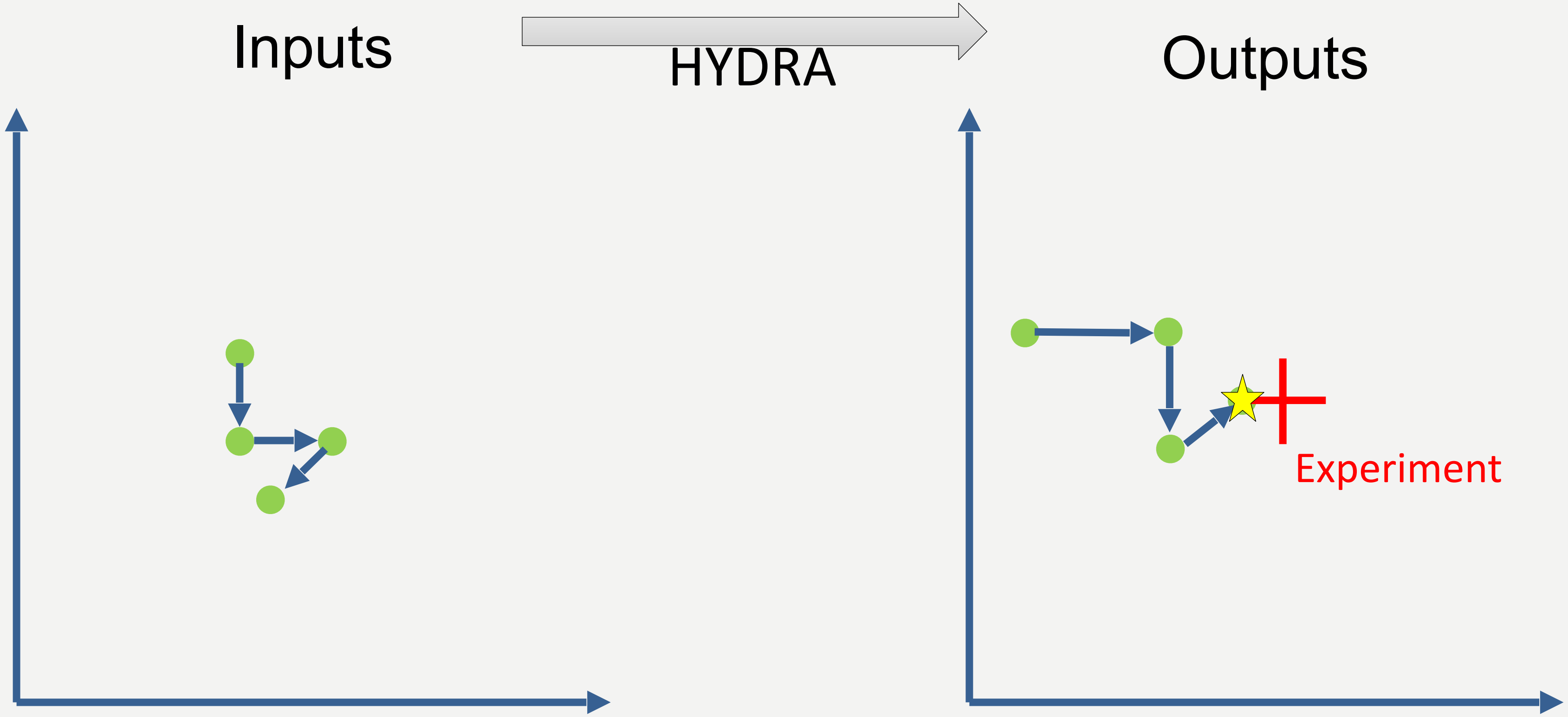
- Inferring unknown simulation parameters using experimental measurements
- Often assumes simulator has low error, but inputs are not well known
- Commonly used to understand the data after the experiment (post-shot analysis)

Model Calibration

- Use experimental data to adjust simulation predictions
- Often assumes simulation inputs are known, but simulator has error
- Commonly used to predict outcome of future experiments

Traditional post-shot analyses do not find full distribution of simulations consistent with experimental observables

Standard Post-Shot



Post-shot: Manual adjustment of simulation inputs until outputs match experiment

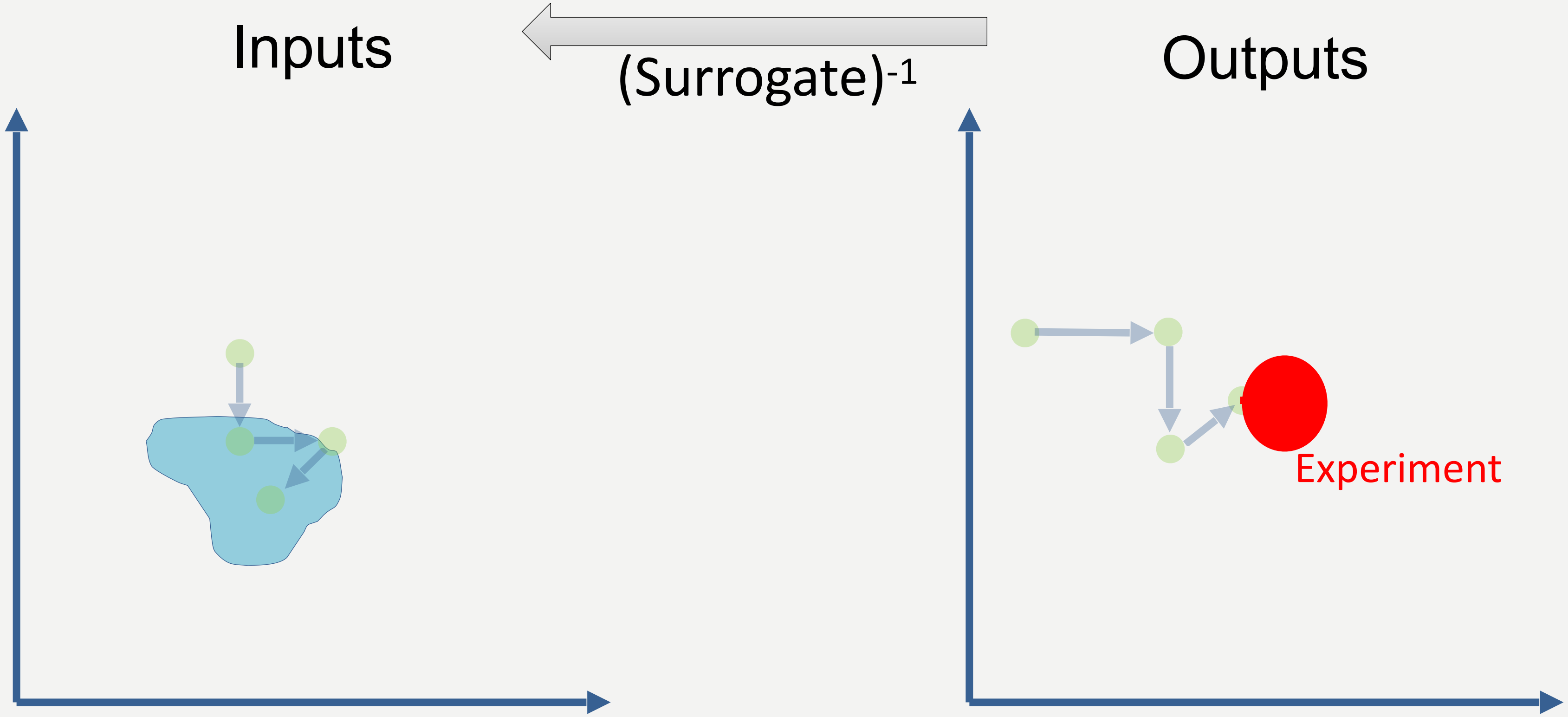
A data-driven approach could improve post-shot analysis for NIF experiments

- Machine learning methods can augment NIF post-shot analysis
  - Inverse models infer **distributions** of inputs that are most consistent with experimental observables
  - Auto-encoders enable us to match dozens of observables simultaneously to better constrain our simulations

Goal: Find the set of physics hypotheses that explain the experimental observations

An efficient alternative to manual searching is to train inverse models

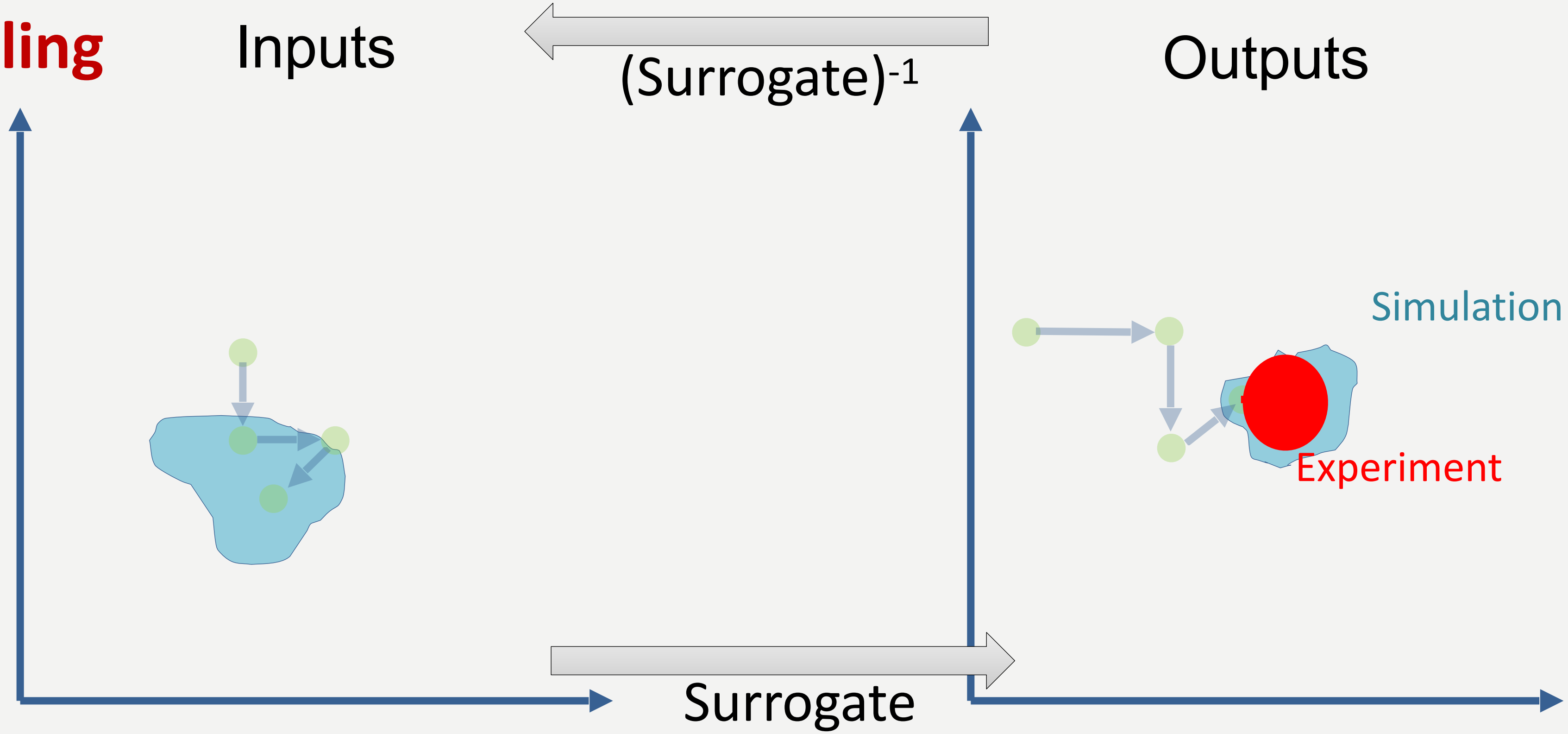
Inverse Modeling



Inverse modeling: Infer *distribution* of simulations that closest match experiment

Agreement between simulation and experiment can be quantified by comparing post-shot simulation outputs to the experiment

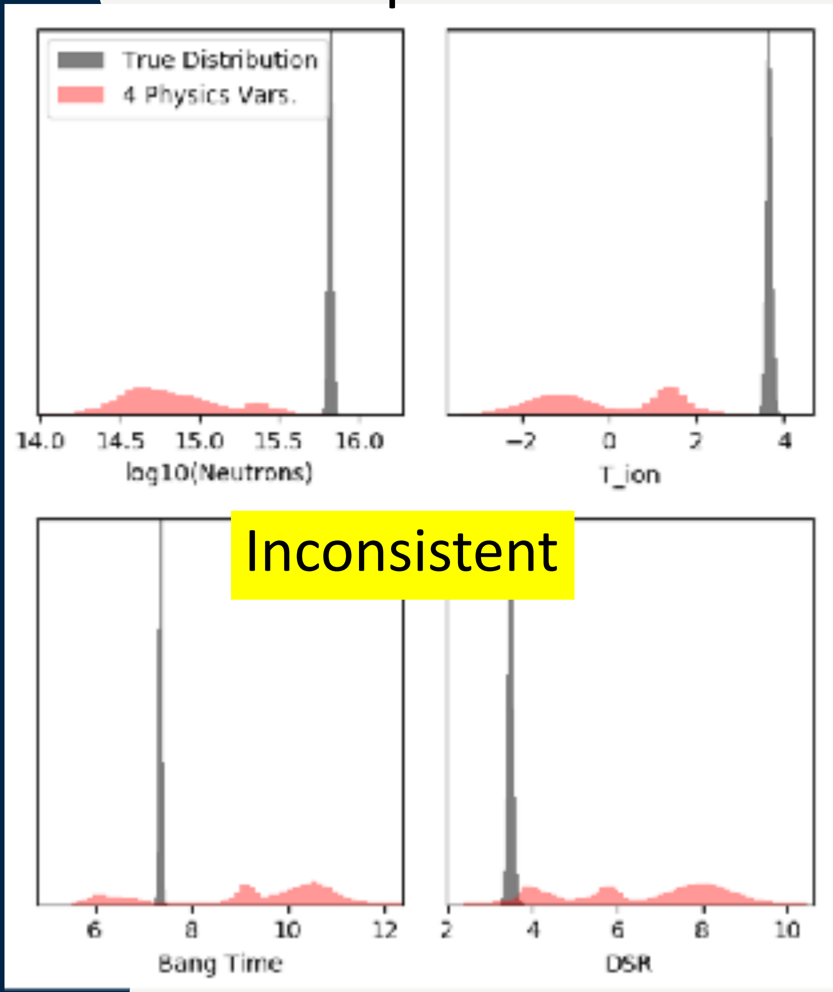
Inverse + Forward Modeling



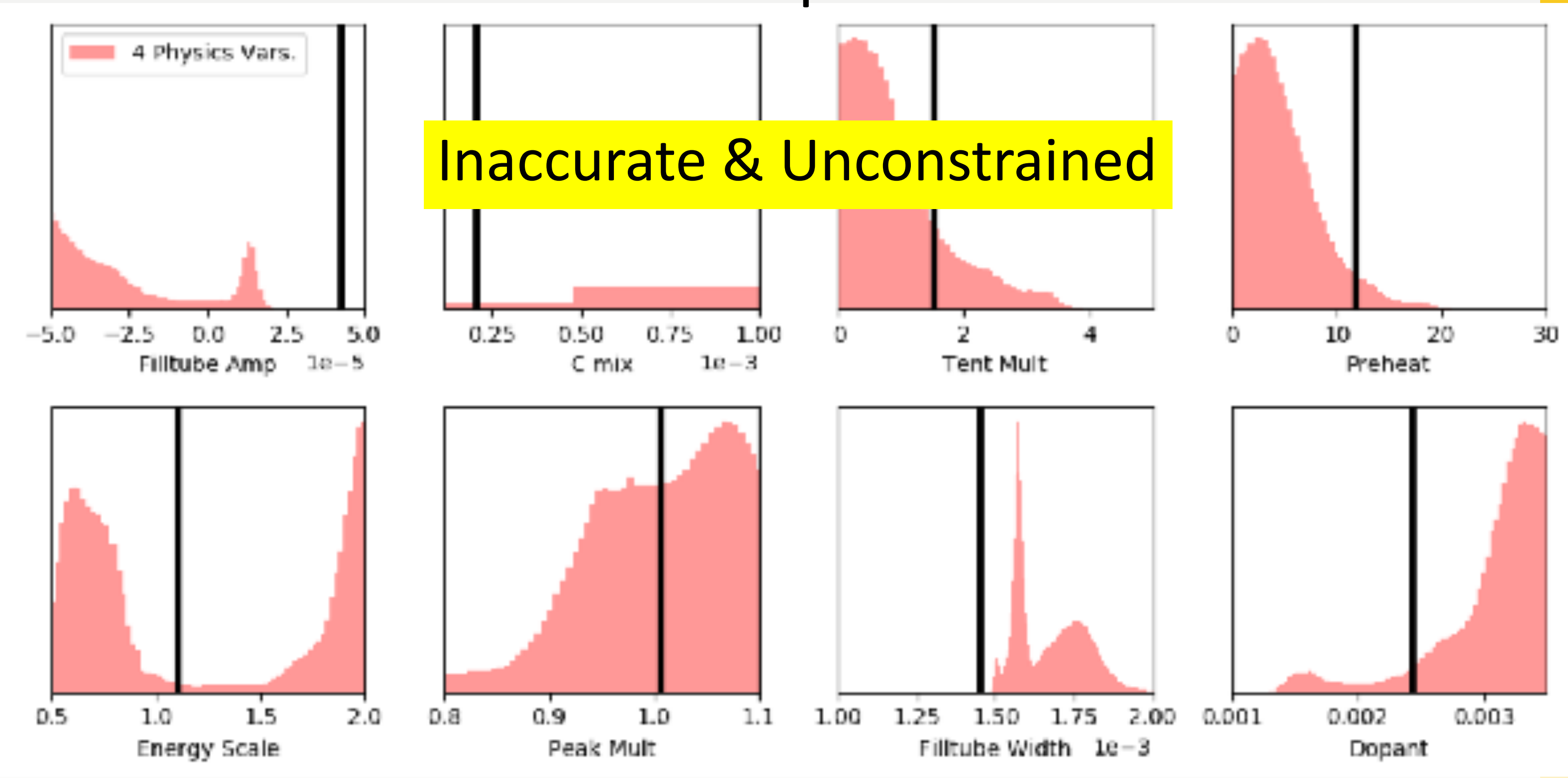
Quantify *how well* closest simulations match experiment

Matching only a few observables does not provide an accurate inverse model or constrain hypotheses

Outputs

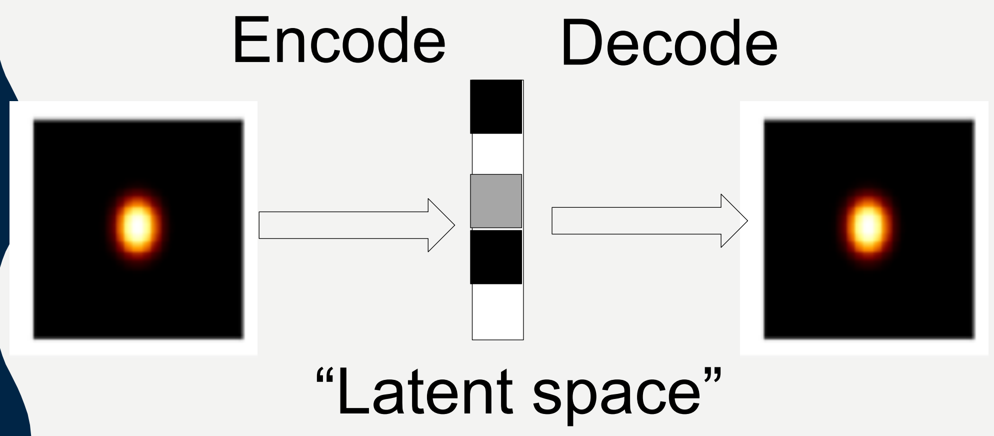


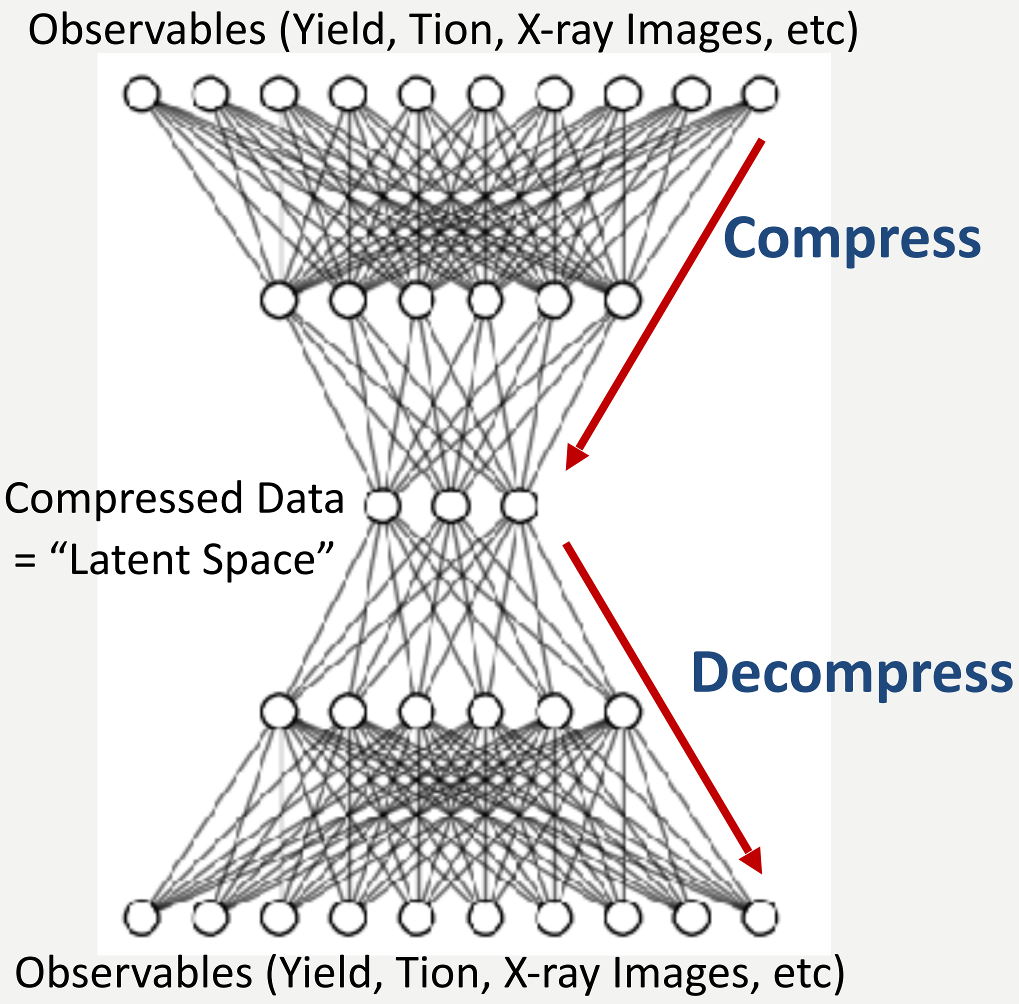
Inferred inputs



For the models to be accurate and constrain our hypotheses, we need to match *more* observables

Auto-encoders enable us to match dozens of observables simultaneously





Can we find the best post-shot simulations by matching a large collection of diagnostics?

Train models on database of 60k 2D HYDRA simulations that span an 8D space:

<u>8 Inputs</u>:

- Added heat to fuel (preheat)
- C mix
- Energy scale
- Peak drive
- Tent amplitude
- Filltube width & amplitude
- Dopant

<u>45 Outputs</u>:

- Yield
- Bang time
- Temperatures
- Neutron spectra moments, DSR (4 lines of sight)
- High energy x-ray yields

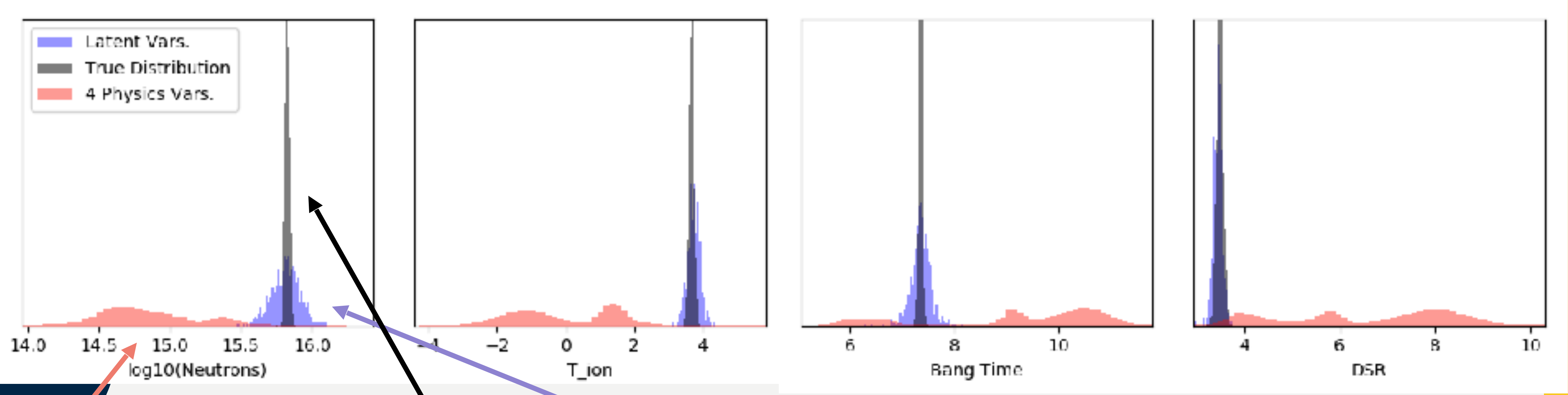
Latent Space (45 auto-compressed observables) ⇄ Simulation Inputs

Yield, Tion, bangtime, DSR ⇄ Simulation Inputs

Does matching all 45 observables produce more accurate post-shots than matching our favorite 4?

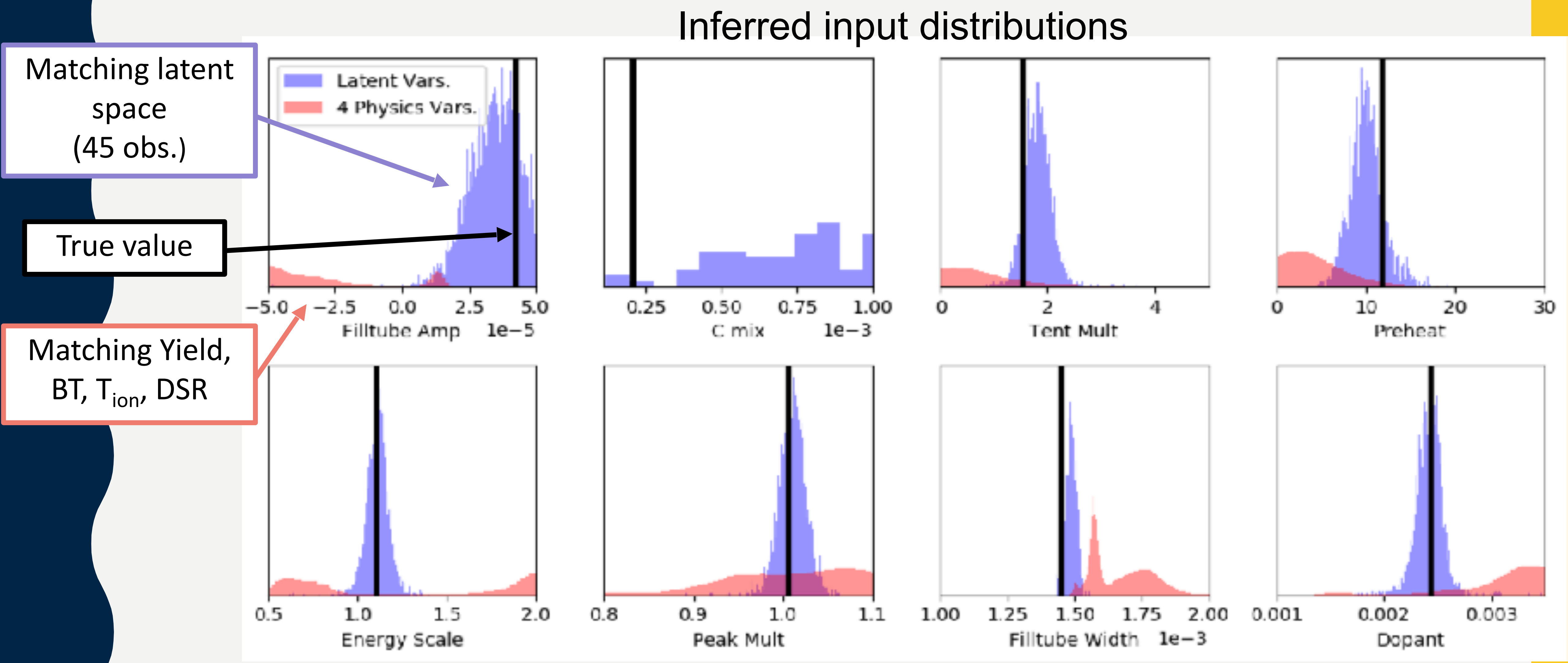
Latent space inference produces a good fit to the experimental data

Predicted output distributions



"Bayesian analysis with deep jointly-informed neural networks" **Humbird**, Peterson, McClarren, Statistical Analysis and Data Mining, 2018 (in revisions).

Latent space inference better constrains our simulation inputs



Neural network post-shot analysis finds the set of hypotheses that explain experimental observations

- Auto-encoders and DJINN enable us easily find all of the simulations in our design space that are consistent with several dozen experimental diagnostics
- Applying to NIF data is challenging -- hypotheses included in our database are not able to explain all of the experimental data

How can we create predictive models if we cannot find accurate post-shot simulations?

DJINN enables efficient parameter inference and model calibration tasks for merging simulation and experimental data

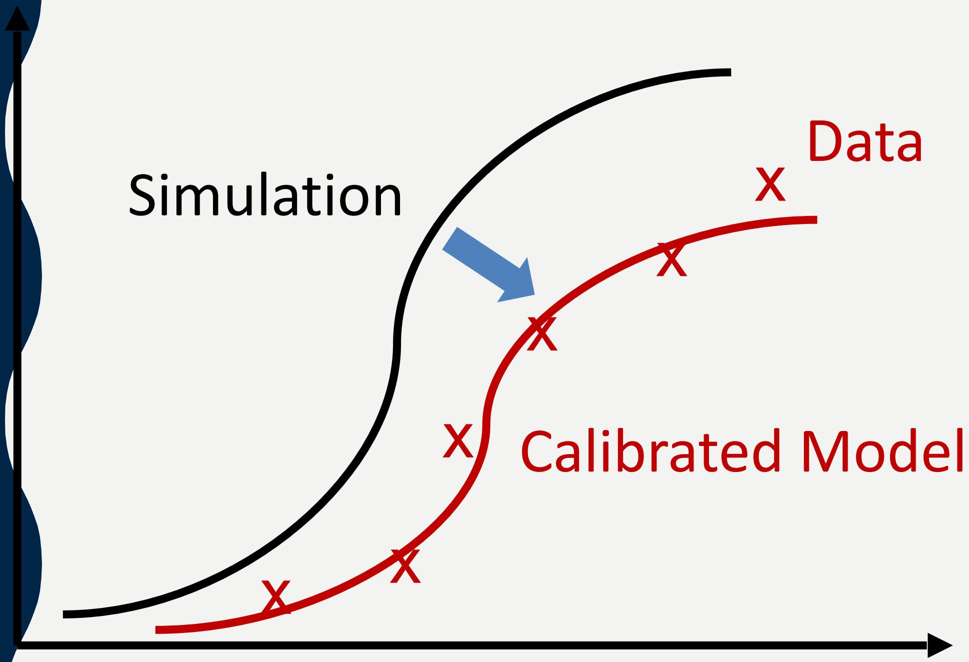
Parameter Inference

- Inferring unknown simulation parameters using experimental measurements
- Often assumes simulator has low error, but inputs are unknown
- Commonly used to understand the data after the experiment (post-shot analysis)

Model Calibration

- Use experimental data to adjust simulation predictions
- Often assumes simulation inputs are known, but simulator has error
- Commonly used to predict outcome of future experiments

DJINN enables efficient parameter inference and model calibration tasks for merging simulation and experimental data



“Bayesian calibration of computer models” Kennedy & O’Hagan (2001)

Model Calibration

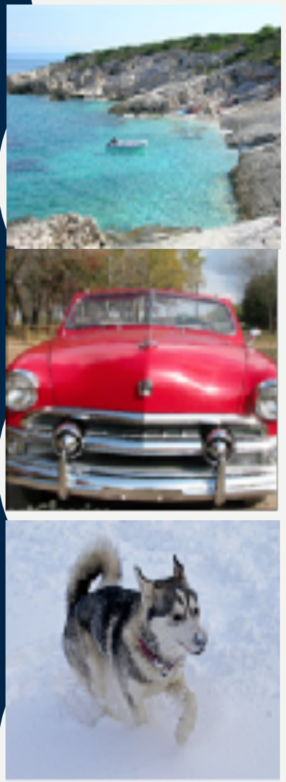
- Use experimental data to adjust simulation predictions
- Often assumes simulation inputs are known, but simulator has error
- Commonly used to predict outcome of future experiments

“Transfer learning” is a popular technique in the machine learning community

Ocean

Car

Dog



Large image database

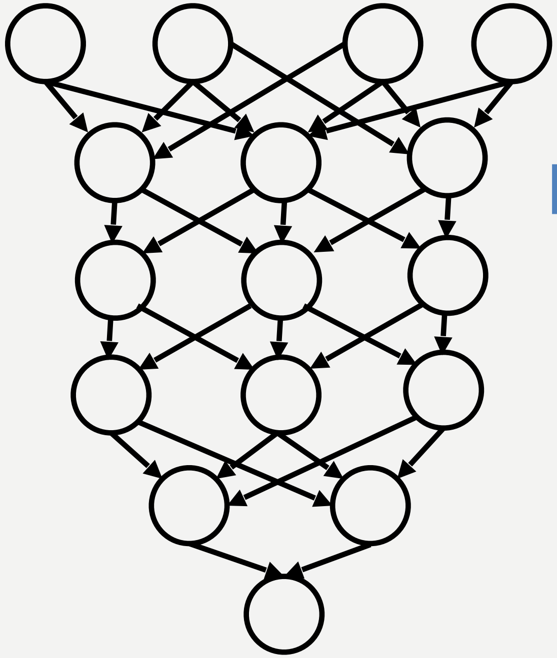
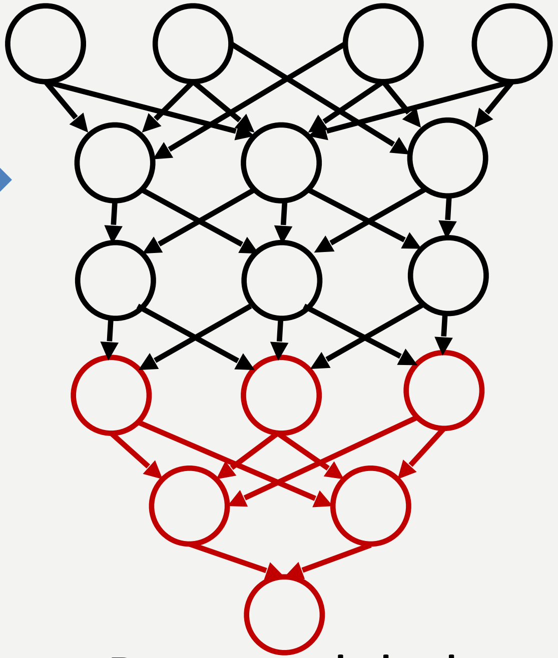


Image label

Retrain to solve different, but related task

Small NIF optics database



Damage label

Thistle

Pansy

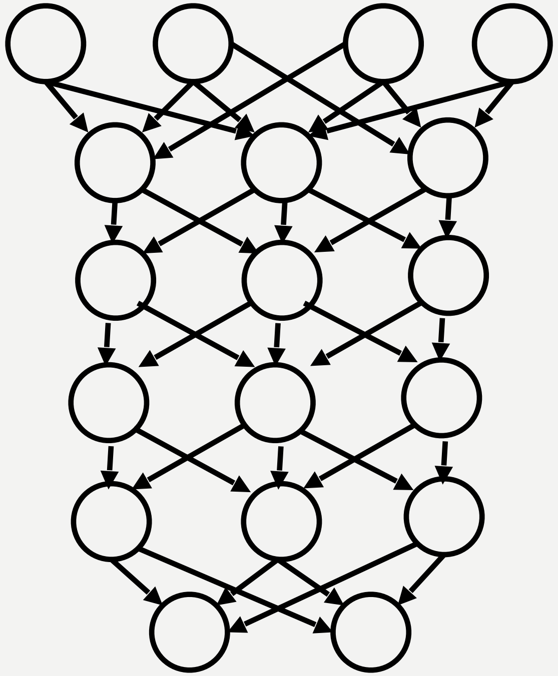
Scratch

Can transfer learning be used to “transfer” between simulations and experiments?

DJINN is used to make more predictive models of ICF experiments via "transfer learning"

Train DJINN on large database of cheap simulations

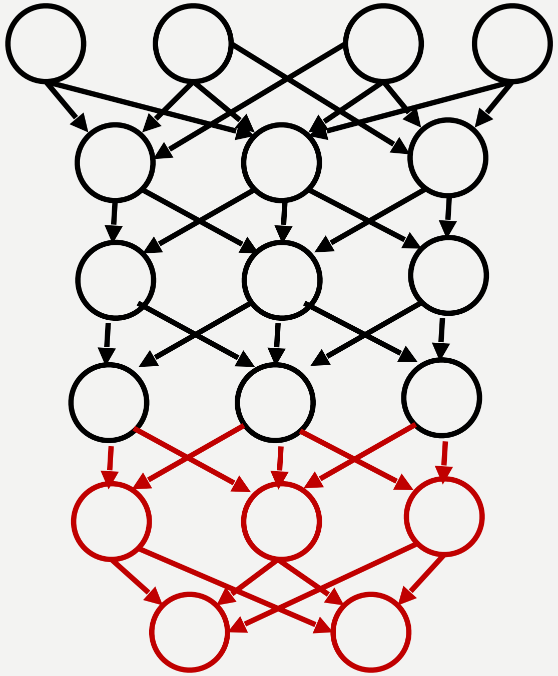
Simulation Inputs



Simulation Outputs

Freeze all but the last layers of the network, retrain on sparse, expensive data

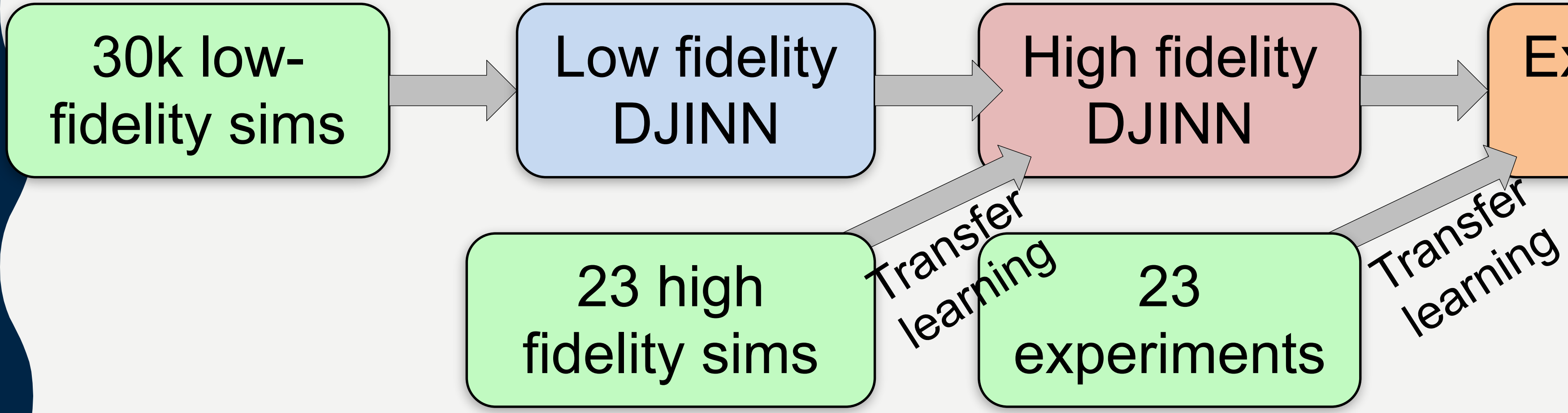
Experiment Inputs



Experiment Outputs

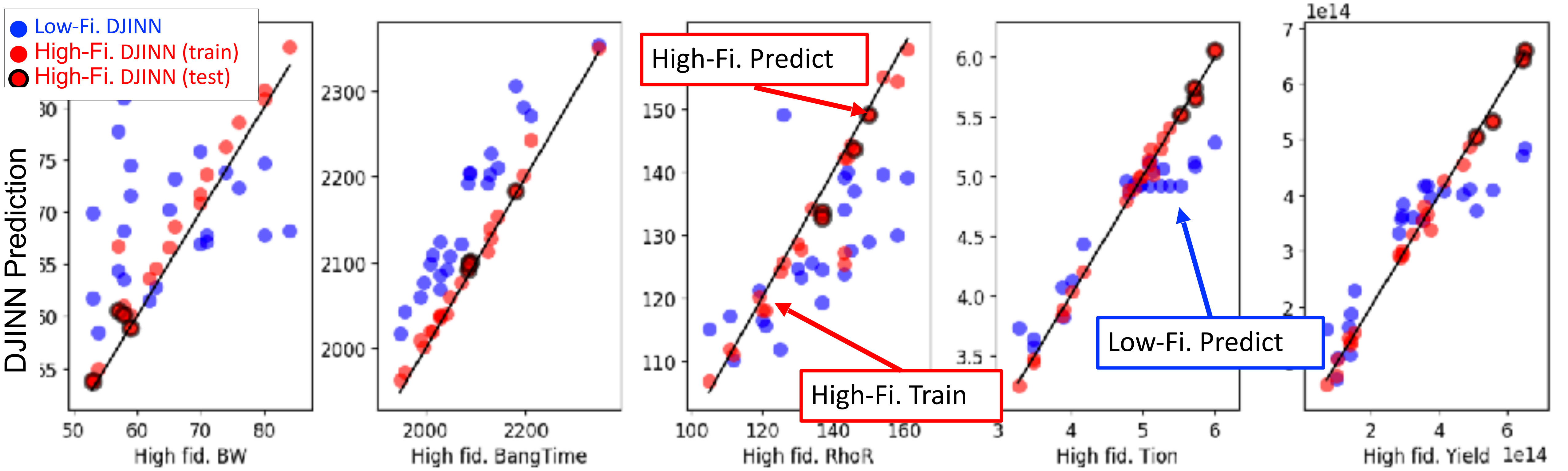
Can DJINN+TL predict future Omega experiments with higher accuracy than simulations?

- The data* includes:
  - 30k 1D LILAC simulations (no CBET, flux-limited thermal diffusion), 10 min runtime
    - Spans a 9D input space with varying laser pulse & capsule dimensions
  - 23 experiments
    - 1D High-fidelity simulations with the 19 scalar outputs (high fidelity with FP_EOS, CBET, non-local electron transport, etc), 8 hour runtime
    - Experimental measurements of yield, bang time, Tion, rhoR, burnwidth



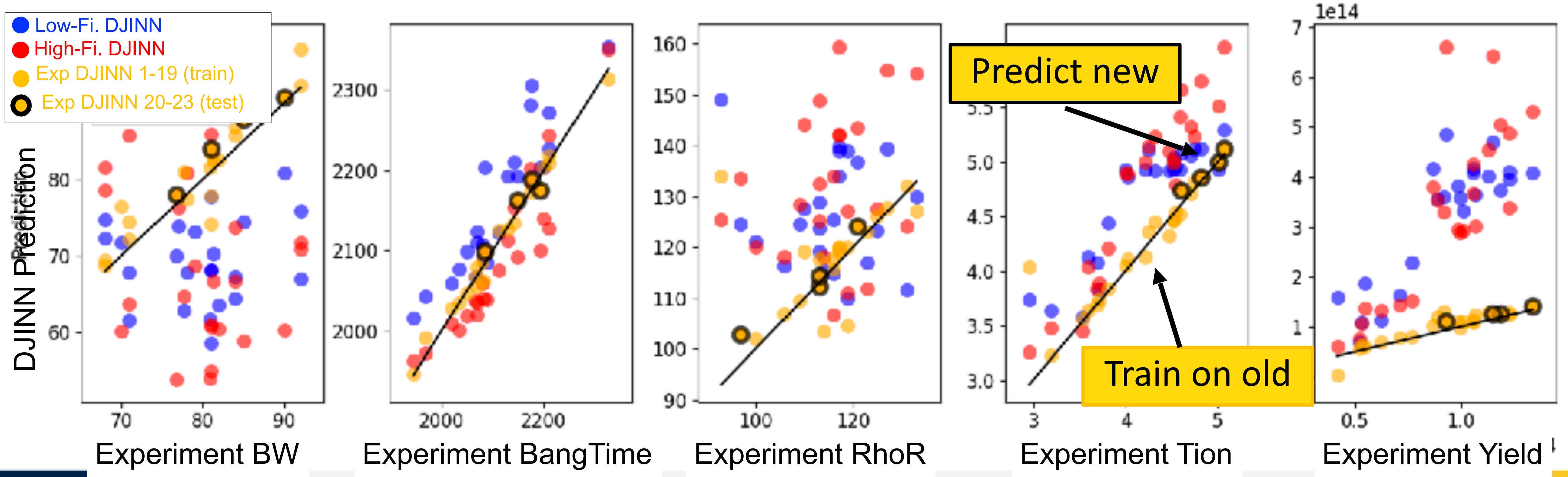
*Data provided by Varchas Gopalaswamy & Riccardo Betti

DJINN+TL: predict high-fidelity simulations with low computational cost



High-fidelity BW High-fidelity BangTime High-fidelity RhoR High-fidelity Tion High-fidelity Yield

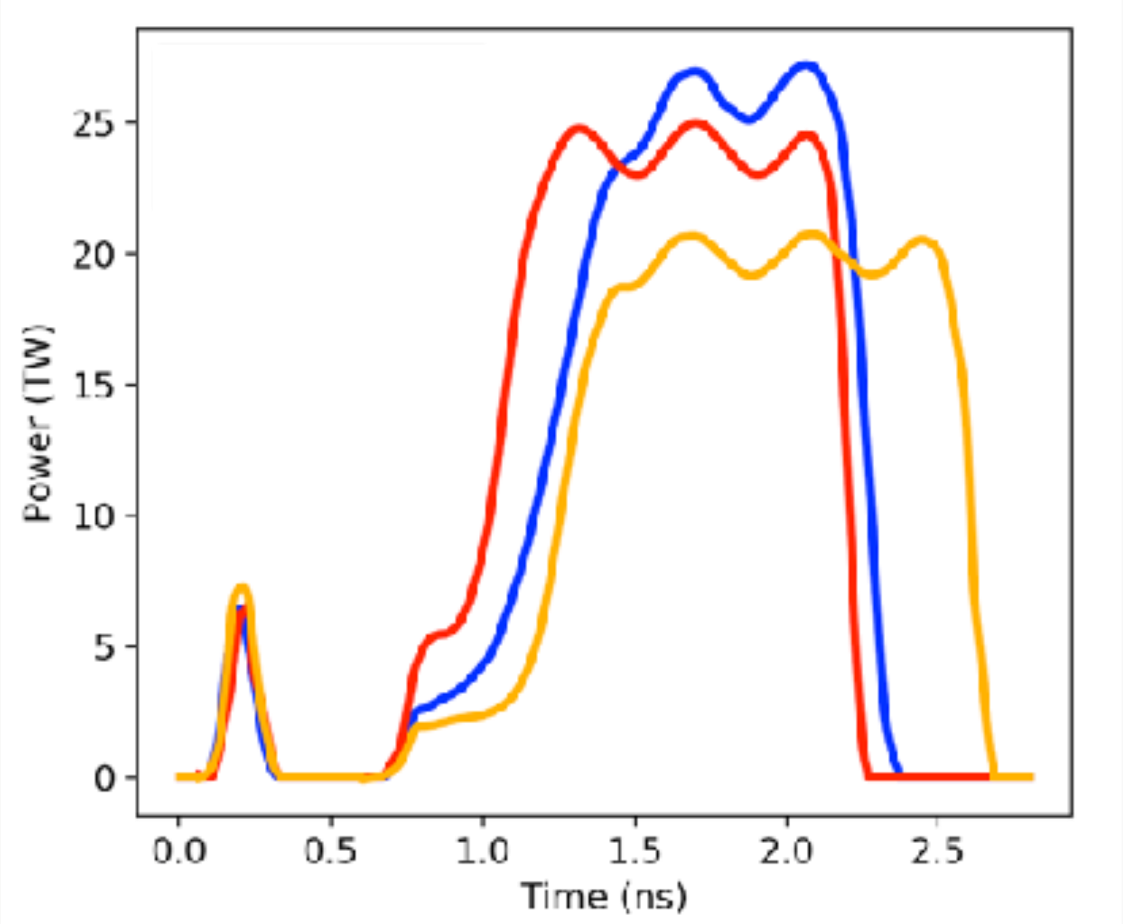
DJINN+TL: *more predictive* of future Omega experiments than simulations



Experiment DJINN models can predict future Omega experiments

Each model suggests a different optimal* implosion

DJINN: Optimal Laser Pulses



Optimal Capsules

*Maximum Yield·$(\rho R)^2$

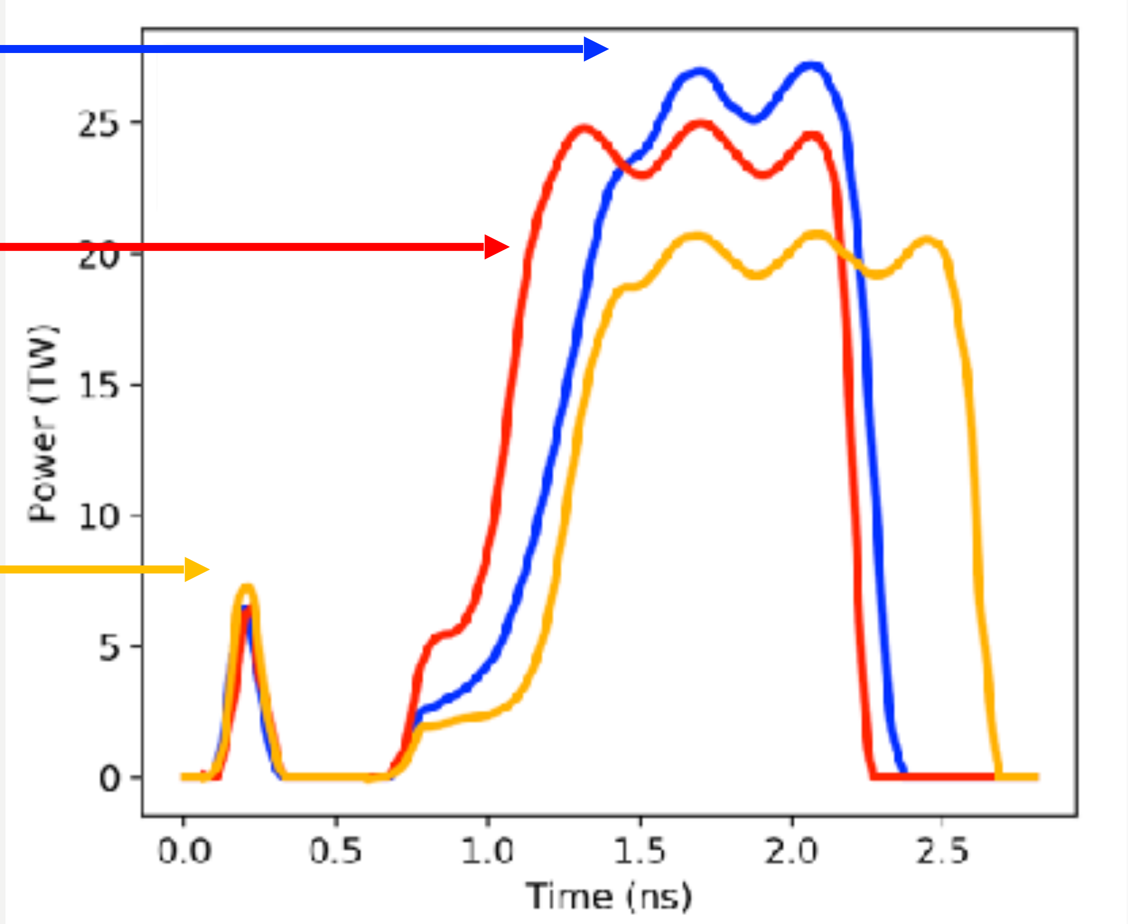
Each model suggests a different optimal* implosion

Max 1D Yield

Fix LPI but maintain velocity

Control hydro. instabilities at higher ρR

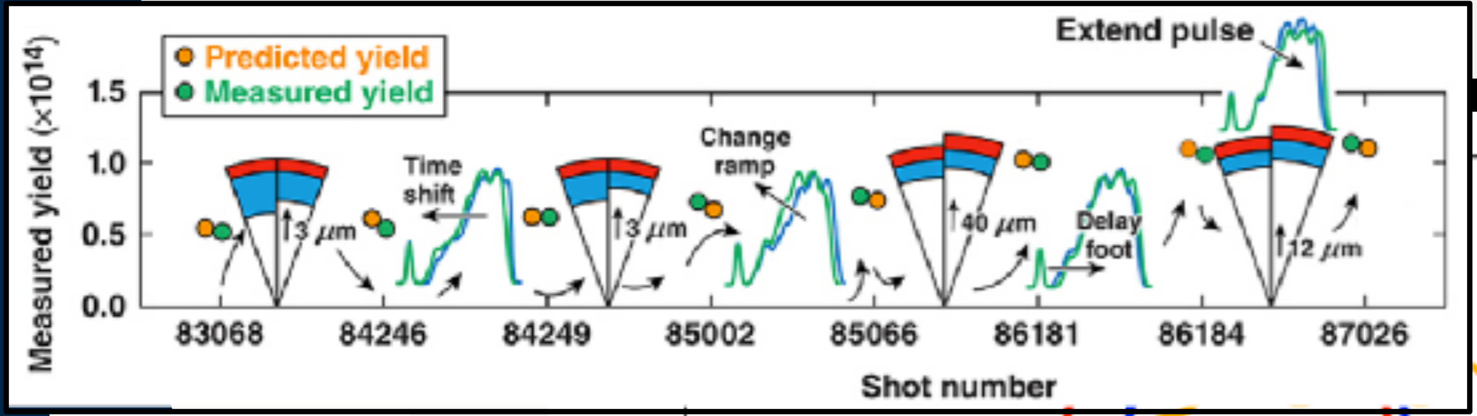
DJINN: Optimal Laser Pulses



Optimal Capsules

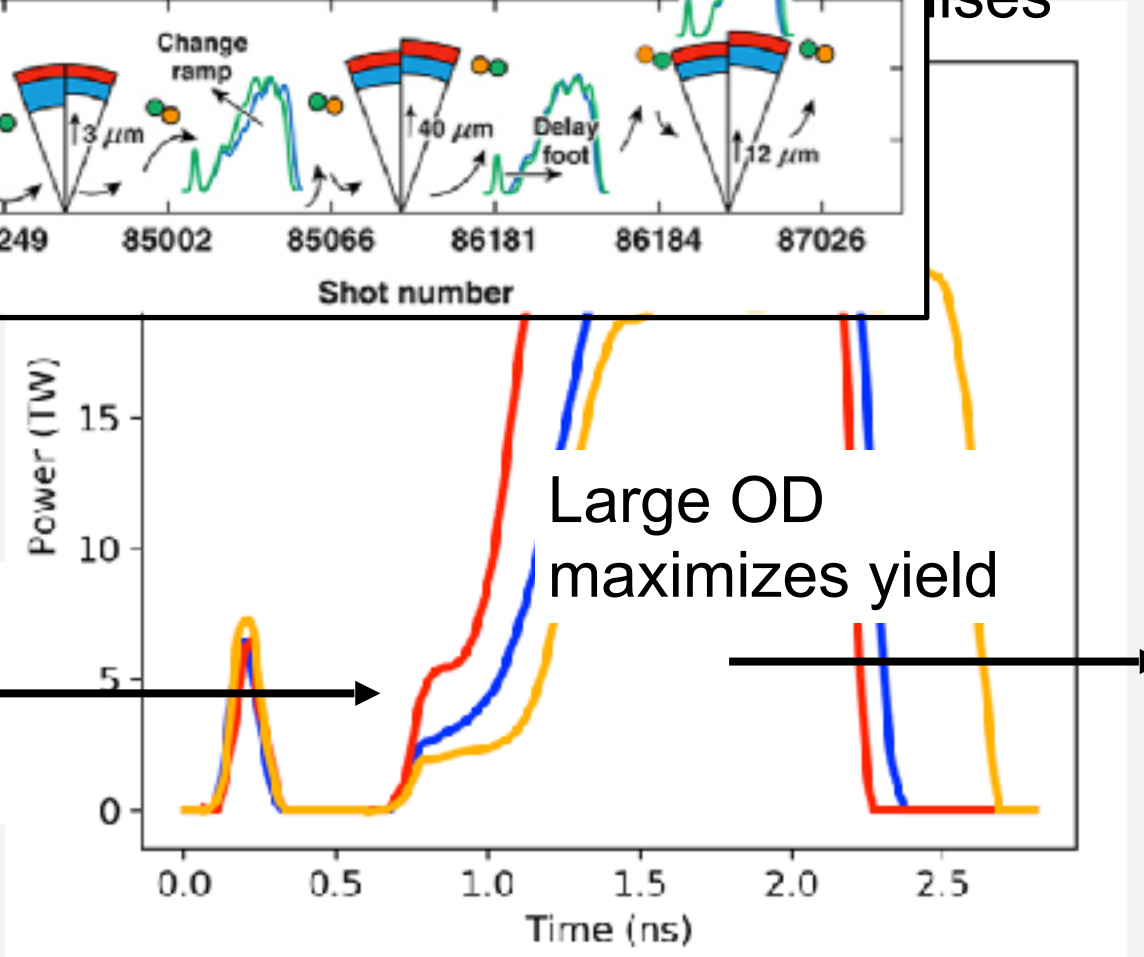
*Maximum Yield·$(\rho R)^2$

DJINN optima are consistent with physics-guided iterations



*Betti et al, 2018

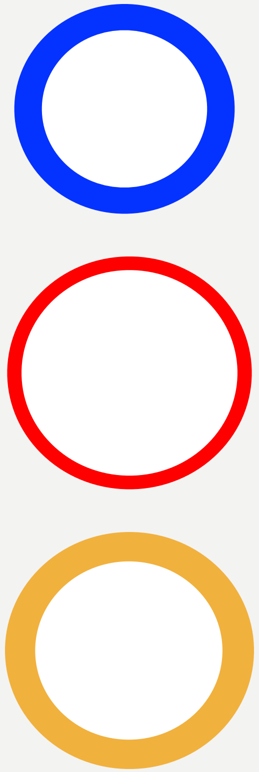
lses



Large OD maximizes yield

Adjusting picket and foot improves ρR

Optimal Capsules



Low-fidelity DJINN

High-fidelity DJINN

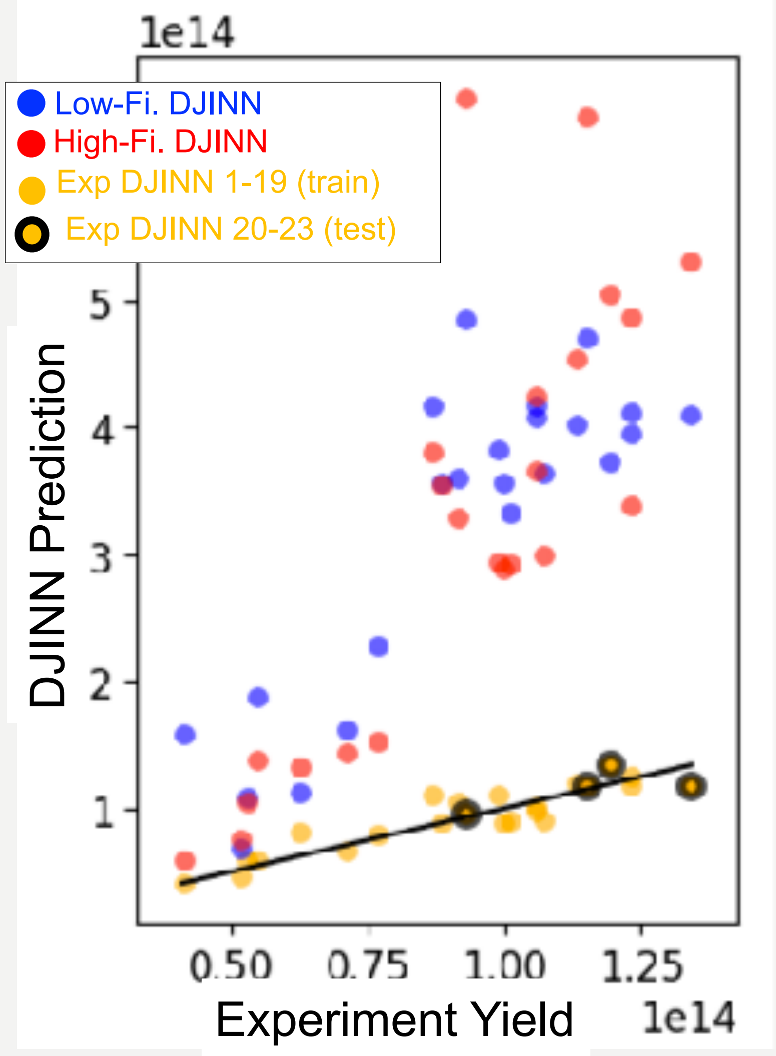
Experiment DJINN

*Maximum Yield$\cdot(\rho R)^2$

DJINN+TL is a powerful, novel method for model calibration

- Transfer learning is a powerful nonlinear calibration technique
- Enables creation of high-fidelity surrogates with low computational cost
- Produces models that are predictive of Omega experiments

Transfer learning creates models that are more predictive than simulations alone



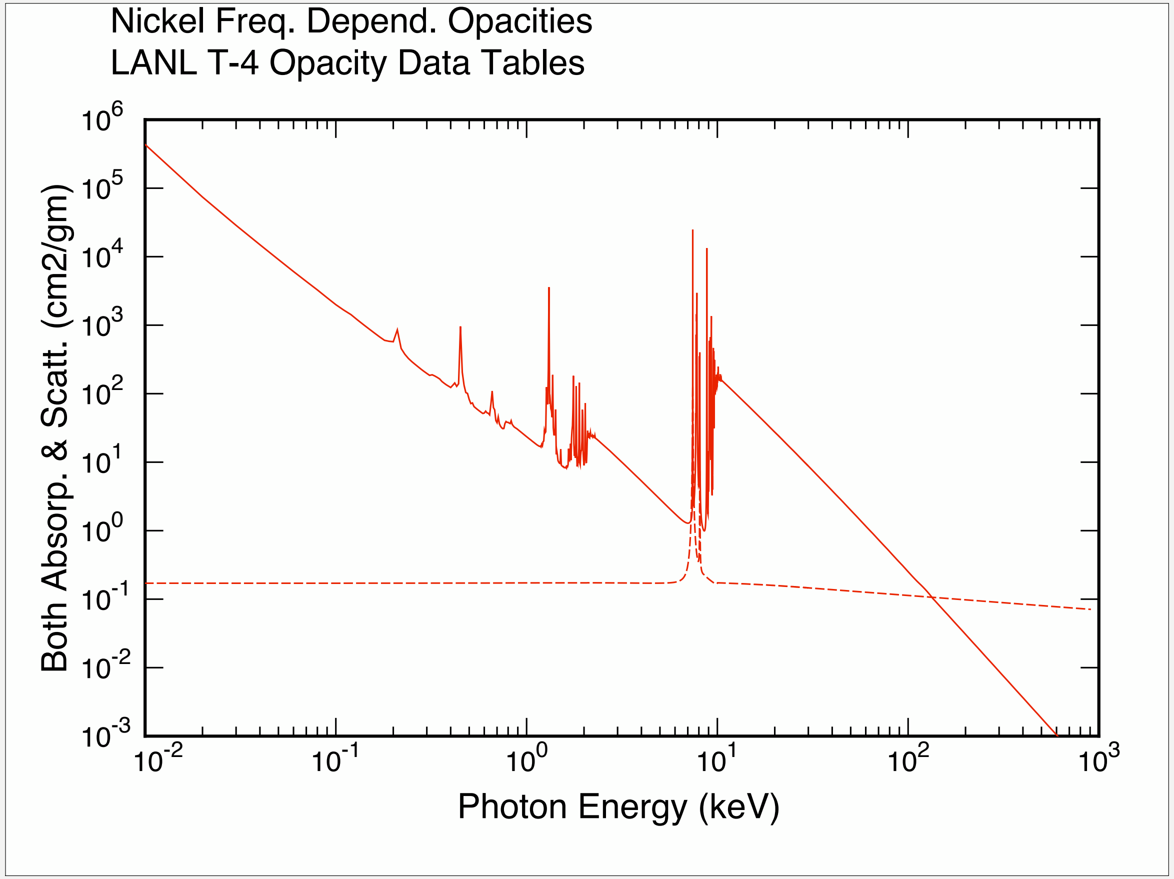
Designing Opacity Experiments with Machine Learning

McClarren RG, Tregillis IL, Urbatsch TJ, Dodd ES. High-energy density hohlraum design using forward and inverse deep neural networks.

Physics Letters A. 2021 Feb 22:127243.

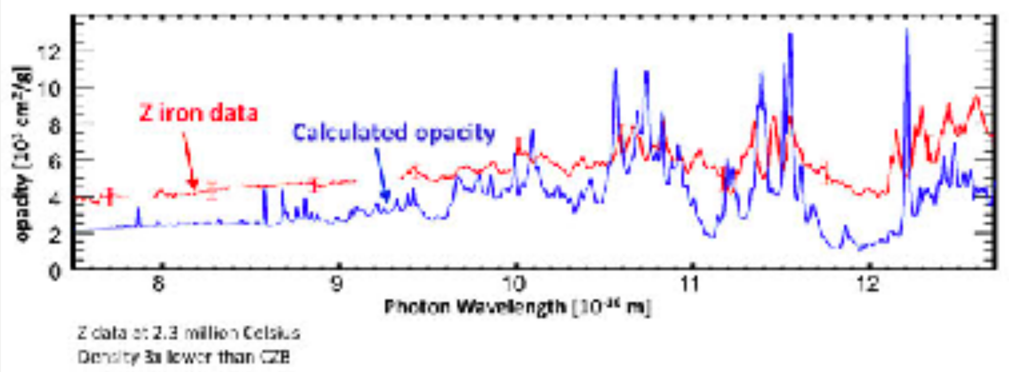
Opacities are key ingredients in high-temperature, participating media radiative transfer simulations

- The opacity of a material gives the strength of the coupling between thermal radiation and matter.
- It is a function of the element, temperature, density, and the frequency (energy) of the radiation.
- In this figure we are looking at the specific opacity of nickel at a given temperature.
  - The absorption opacity (solid curve) has features corresponding to atomic transitions.
  - The scattering is a much simpler function of the radiation energy.
- The opacity indicates how strongly a material will absorb radiation and how well it radiates in equilibrium.



Recent Measurements indicate that our theoretical understand of opacities is inadequate

- The amount of iron in the sun is an important factor in astronomers' estimate of how stars evolve.
- There is a discrepancy between the sun's observed behavior and that predicted by standard models.
- Experiments on the Z-machine at Sandia National Labs indicated that the opacity for iron is 30-400% higher than atomic physics predicts.
- The Z-machine produces high energy density conditions by passing >20 MA through a wire array.
- One researcher put it to me this way: "Either we know nothing about the sun or nothing about atomic physics."



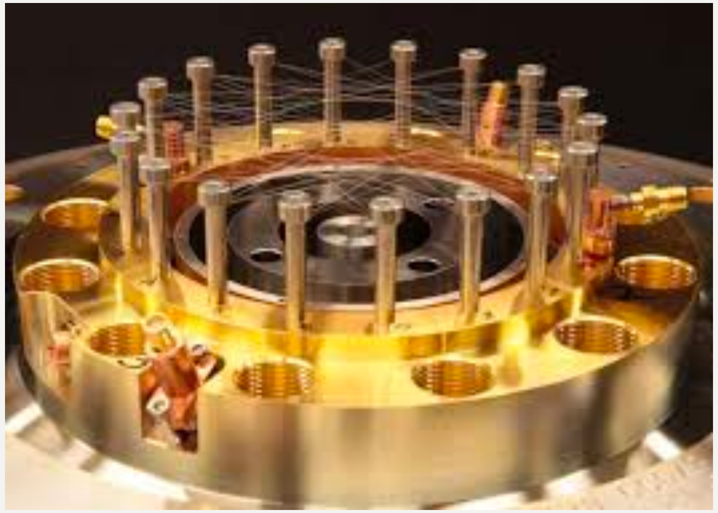
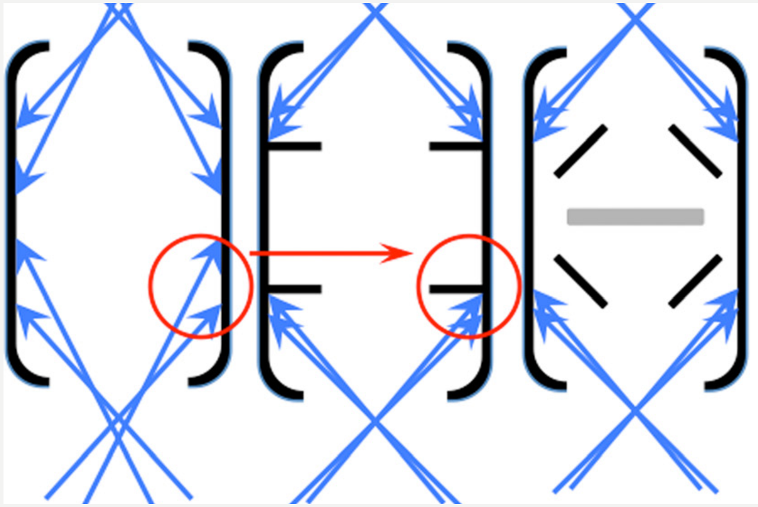
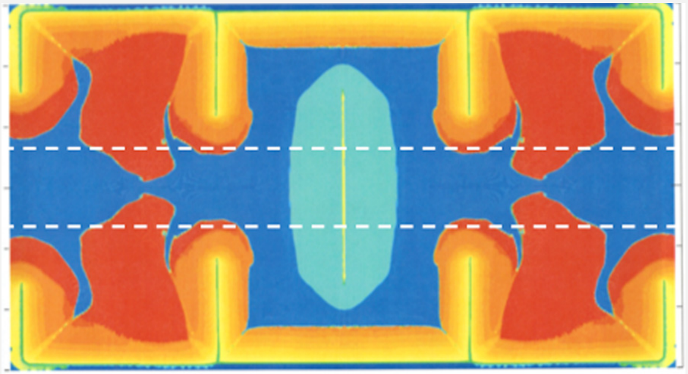


Image credits: https://www.sandia.gov/news/publications/labnews/articles/2019/09-13/solar_model.html OSTI.gov

The NIF laser can be used to confirm these findings.

- Rather than using the Z machine, a sample of iron (gray rectangle) can be placed inside a hohlraum.
- The sample heats up due to radiative transfer and then it is probed with a backlighter (bottom right image) to measure the opacity.
- The hohlraum is a bit more complicated because the sample needs to be shielded from the laser hot spots.
- This then makes the evolution of the system complicated by the fact that the hohlraum cannot close before the measurement is taken.





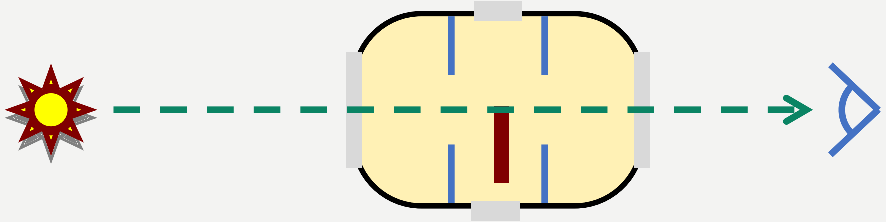
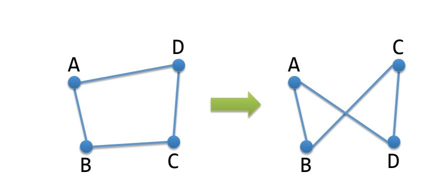


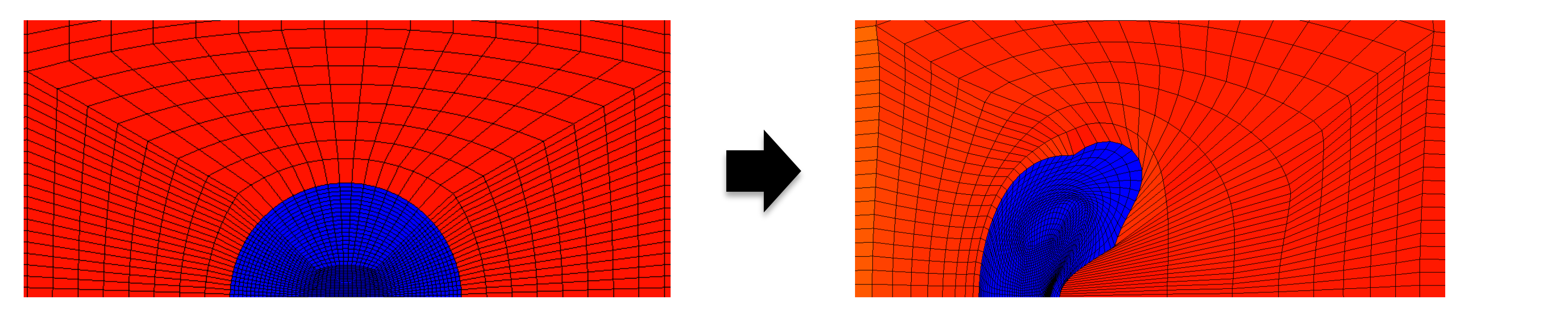
Image credits: Perry, T. S., et al. (2017). High Energy Density Physics, 23, 223–227.
Dodd, E. S., et al. (2018). *Physics of Plasmas*, *25*(6), 063301–11.

Simulations to design these experiments are expensive.

- Because we are dealing with experiments where solids are rapidly turned into an expanding plasma *Lagrangian* methods for fluid flow are commonly used.
- These simulations have the mesh move with the material as much as possible.
    - The problem is that instabilities in the evolution cause the mesh to tangle, leading to negative volumes and causing the simulation to crash.

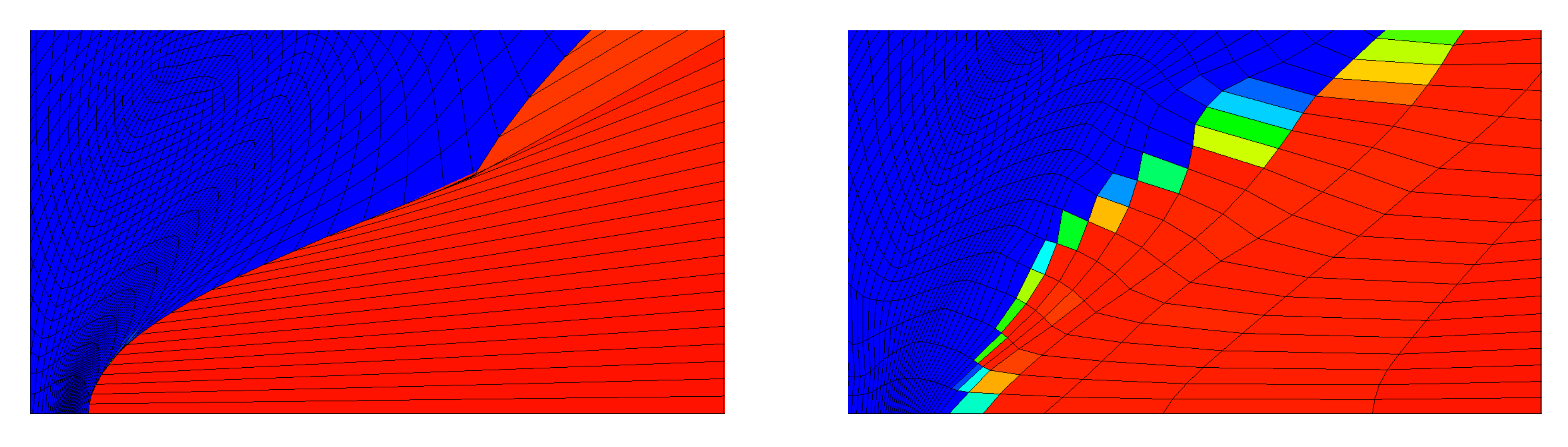


- Therefore, in parts of the simulation the material is allowed to flow through the mesh leading to *Arbitrary Lagrangian-Eulerian (ALE) methods.*
- It takes skill and knowledge of how a simulation should evolve to set parameters for mesh relaxation.
- If the ALE strategy is not properly set, the simulation crashes and the mesh needs to be fixed by hand (possibly every time time).



Figures from LLNL-PRES-660220

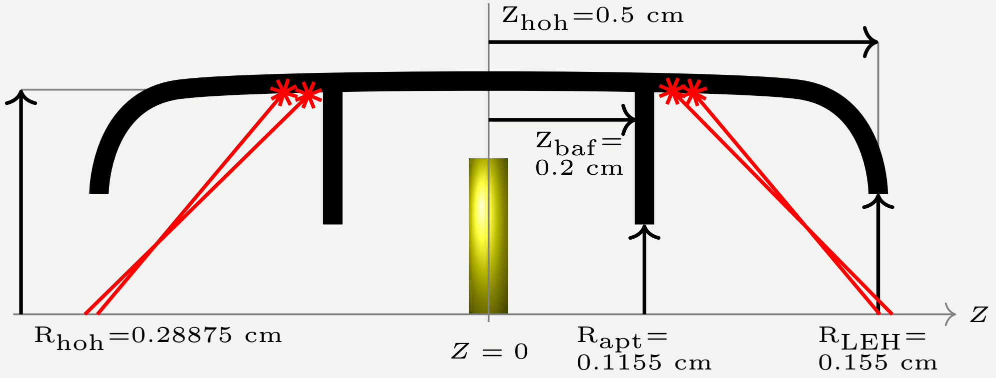
Too much relaxation, however, leads to errors.

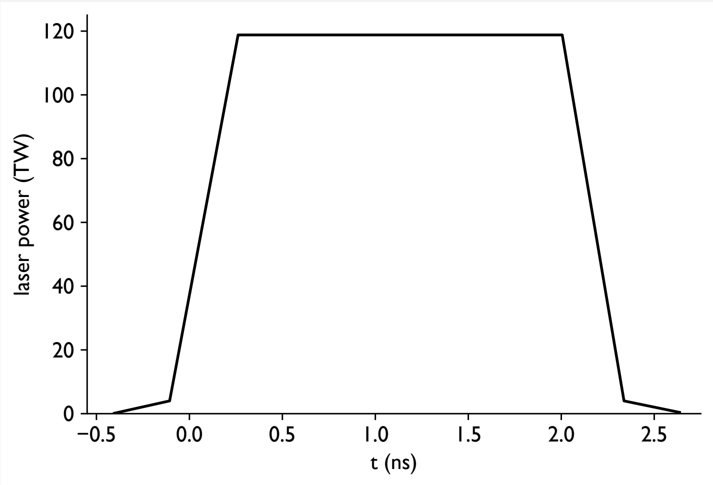


- When the mesh is too constrained, the interface between materials will not be adequately captured and the solution develops numerical errors.

Despite simulation challenges there are a variety of parameters we seek to optimize in our designs.

- To field an experiment to measure the iron opacity we want a hohlraum that can deliver
  - A high temperature for the sample
  - A flat temperature in time
- For the hohlraum there are four parameters we consider that adjust the nominal hohlraum (shown top right):
  - A `scale` parameter that sets the overall size of the hohlraum by scaling the measurements.
  - A `sc_length` (sample chamber length) parameter that varies Z_{baf} while keeping R_{apt}/Z_{baf} constant.
  - An R_{apt} (aperature radius) parameter for scaling the size of the aperture between the sample and laser illumination chambers independent of Z_{baf}.
- Finally, the length of the laser pulse (`pulse_length`) scales the laser pulse length in time but keeps the energy delivered to be a constant 250 kJ.





Laser Pulse Shape

McClarren RG, Tregillis IL, Urbatsch TJ, Dodd ES. High-energy density hohlraum design using forward and inverse deep neural networks. Physics Letters A. 2021 Feb 22:127243.

We then run a set of simulations to try & cover our 4-D design space.

- Considering the range of each parameter from 0.8 to 1.2 we tried to run as many simulations as we could.
- We had an input file for the nominal case (1,1,1,1) with relaxation settings that allowed it to complete.
- Then we tried to run all of the “corners” of the 4-D hypercube and the centers of the faces to see how varying the parameters affected the performance of the hohlraum.
- Several simulations crashed due to mesh tangling and we had 46 simulations complete.
- Here we show all of the 2-D projections of the 4-D space
  - O is a run that completed
  - X is a run that failed
  - The red triangle was a test point (stay tuned)

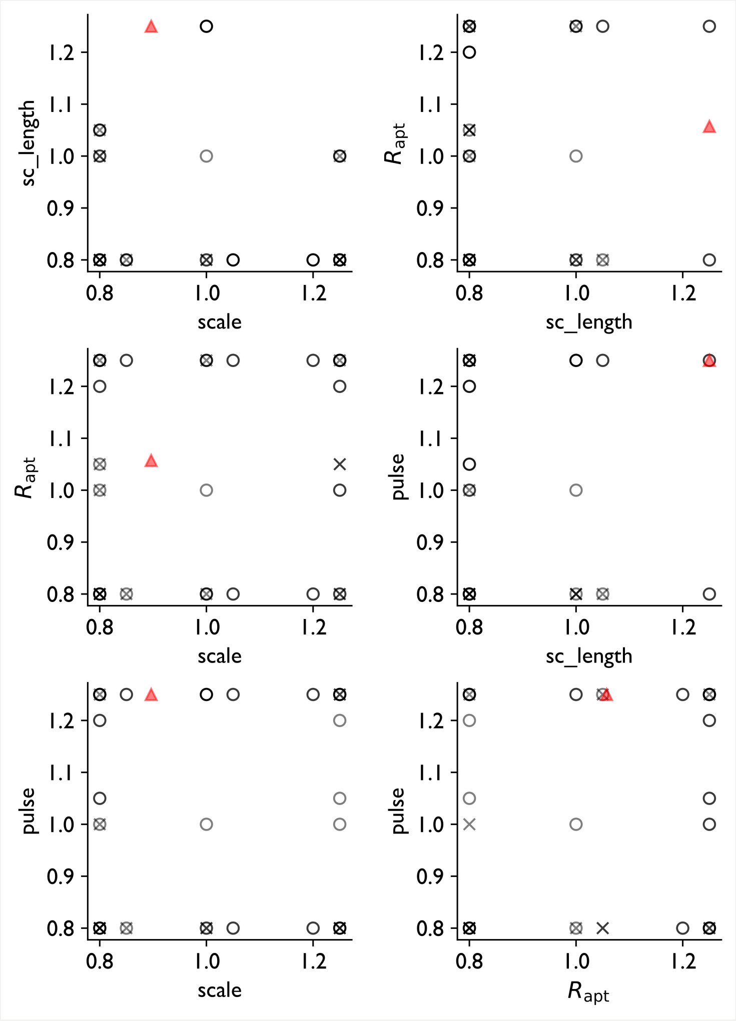
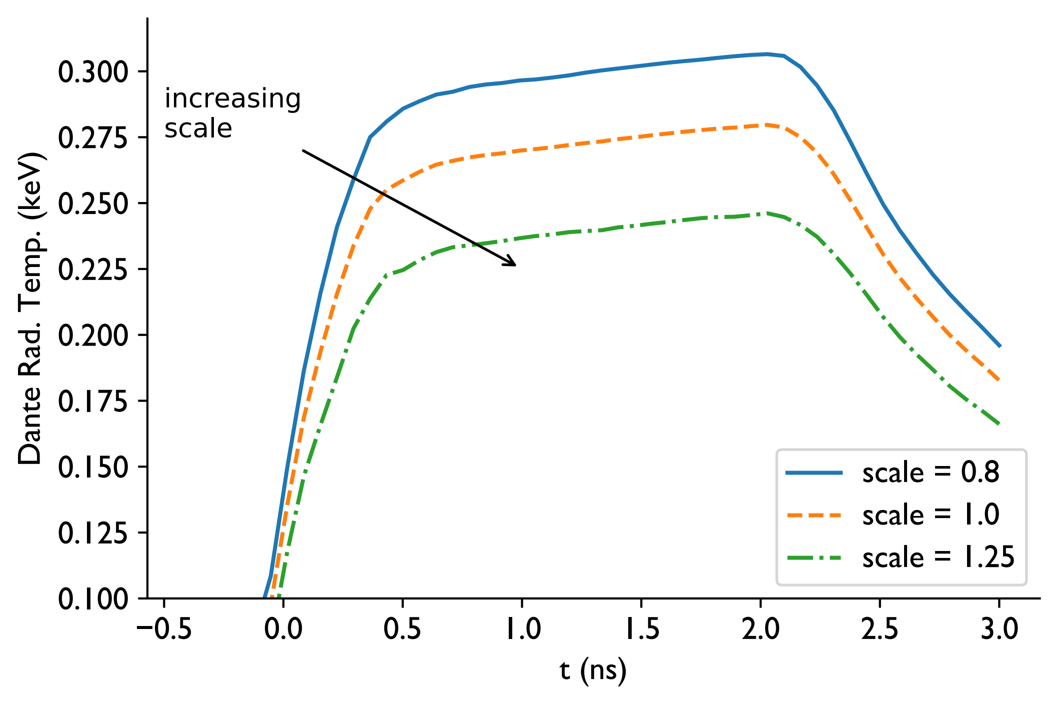


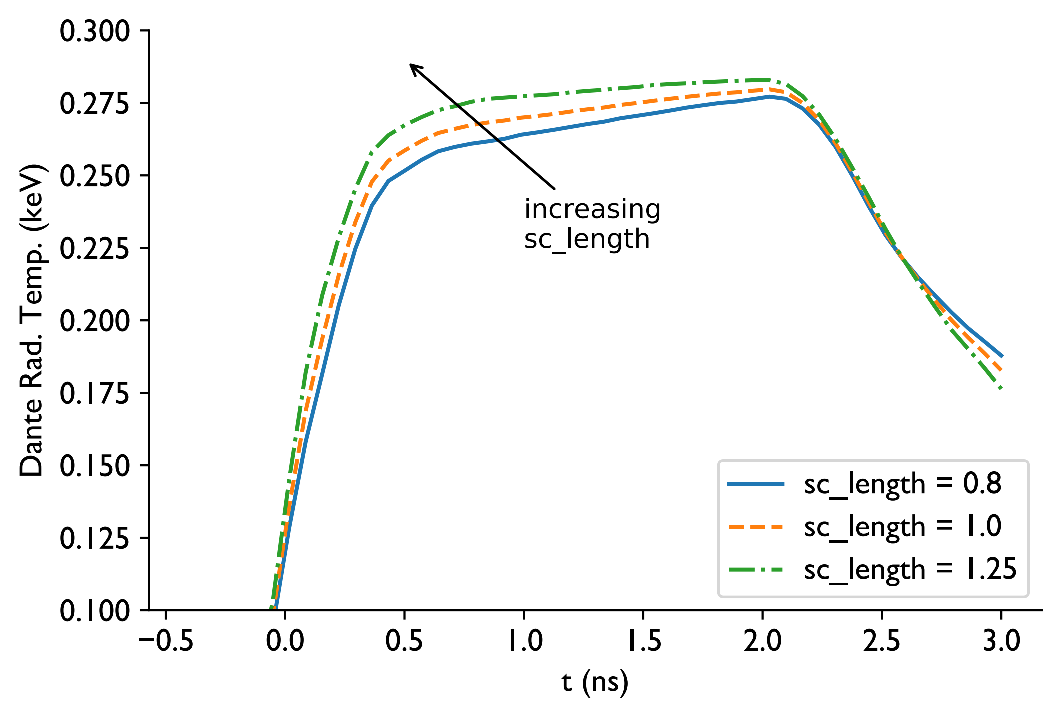
Figure 3: Values of the 4 design parameters in the simulation ensemble. Circles indicate simulations that completed, X’s denote runs that failed.

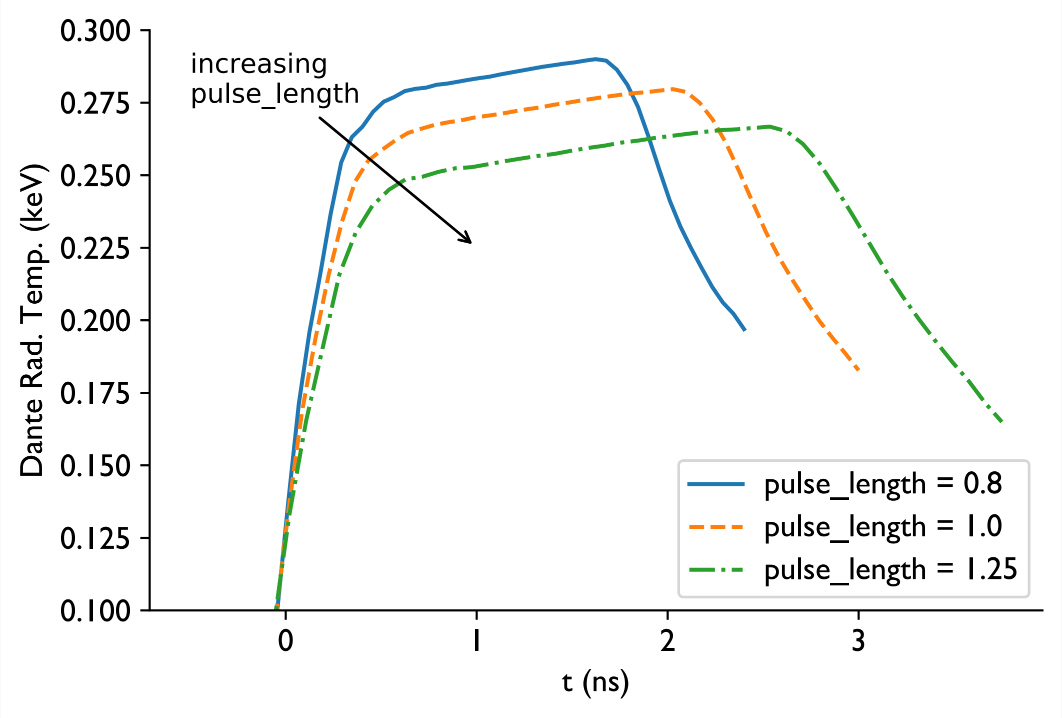
Our output of interest is the time profile of the radiation temperature of the hohlraum.

- Dante is a diagnostic device at NIF that measures the radiative field of an experiment.
- The radiative filed strength is related to the radiation of a blackbody at equilibrium via the "radiation temperature."
- Our simulation code can also predict the response of this diagnostic.
- In a typical simulation there is a rise in the temperature associated with the increasing laser energy.
- This is followed by a slowly varying plateau and a cooling phase occurring after the laser turns off.
  - It is during the plateau that the opacity measurement would be taken.
- From the simulations we observe that a smaller hohlraum leads to a higher temperature.



The sample chamber length can flatten the profile and shorter laser pulses increase the temperature.

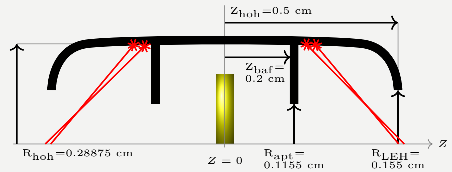




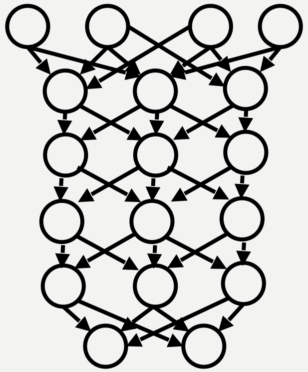
- The Rapt parameter had negligible impact on the Dante temperature profile

To optimize our designs without more simulations, we built a neural network to predict the Dante output.

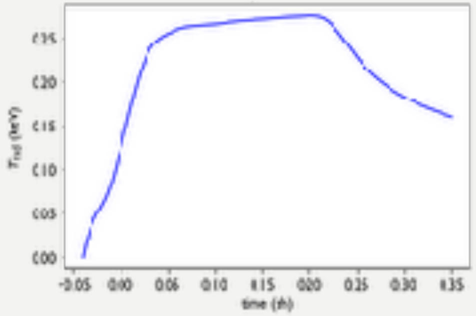
- We recorded the Dante temperature at 50 different time points for each simulation.
- We then trained a feed forward neural network to learn the mapping from the 4 parameters to the temperatures at the 50 times.
  - That is, we input 4 numbers and get out 50 numbers.
- The network has 4 hidden layers:
  - A dense layer with 8 hidden units
  - A dense layer with 100 hidden units
  - A dense layer with 50 hidden units
  - A convolution layer with a kernel size of 3
- This architecture was chosen (after some exploration) with the following principles in mind:
  - Make a preliminary computation on the inputs,
  - Expand the result to a large number of units and then,
  - Map that result smoothly to the output using a convolution.
- We also used dropout to avoid overfitting. This randomly turns off connections during training the model.



Simulation Inputs (4-D)

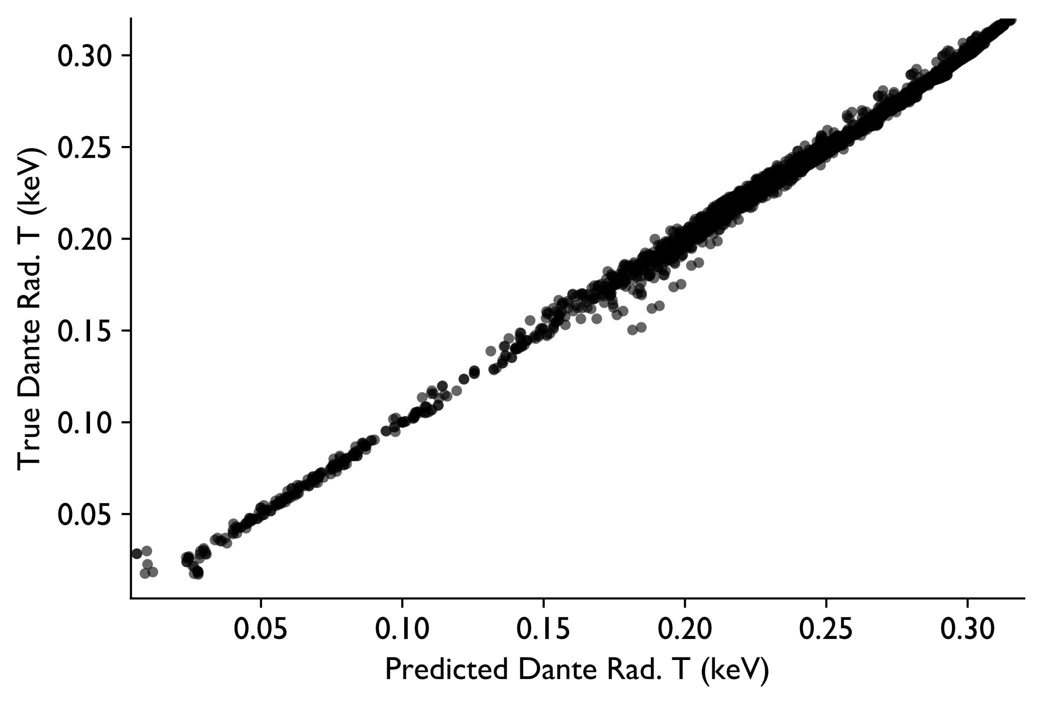


Simulation Outputs (50 time points)



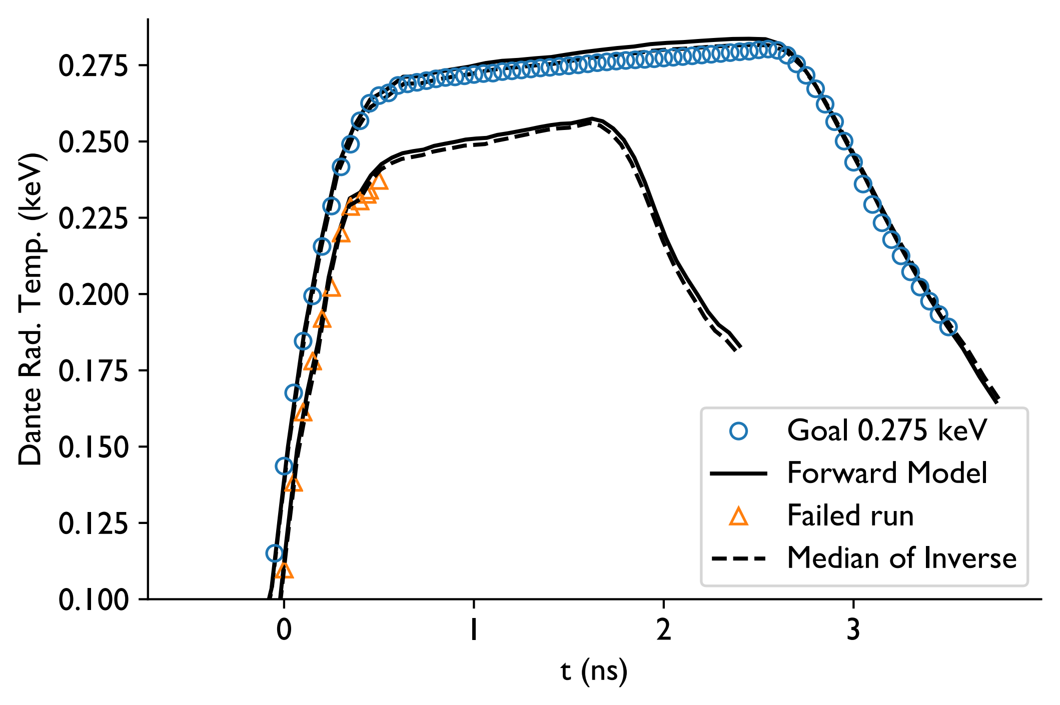
We used Leave-One-Out (LOO) Cross-Validation to test this “forward” model.

- Given the small dataset that we had to work with, we tested our model using Leave-one-out cross-validation
  - Train the NN using 45/46 simulations, and predict the response for the 46th.
- The figure shows the result from applying this across the data set
  - The predicted value from the NN model is the x-position.
  - The true value is the y-position.
  - A perfect model would have all of the points fall on the x=y line.
- We estimate a mean-absolute error from the prediction to be 0.003 keV.
- We call this model a forward model because it maps parameters to experimental measurements.



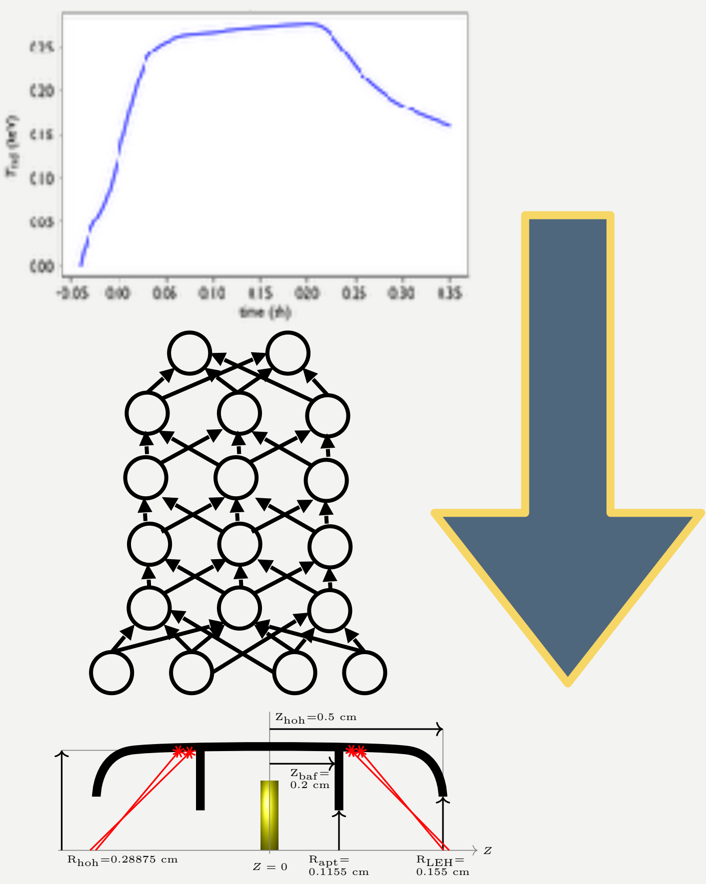
We can use the forward model to predict what the result of a crashed simulation or new simulations would be.

- We took the settings for one of the simulations that crashed due to mesh tangling and used the model to predict the behavior.
- This data was not used to train the model, but we can see that it is accurately predicting the behavior of the solution time that did complete.
- It predicts that the peak temperature occurs before 2 ns.
- We also used the model on a novel set of parameters designed to make the flattest, longest possible plateau at 0.275 keV.
    - This was the red triangle on the parameter plot before.
- We can compare the model to the results of a simulation for this case.
    - A slightly higher temperature is observed in the model.
- What about this “inverse” model?



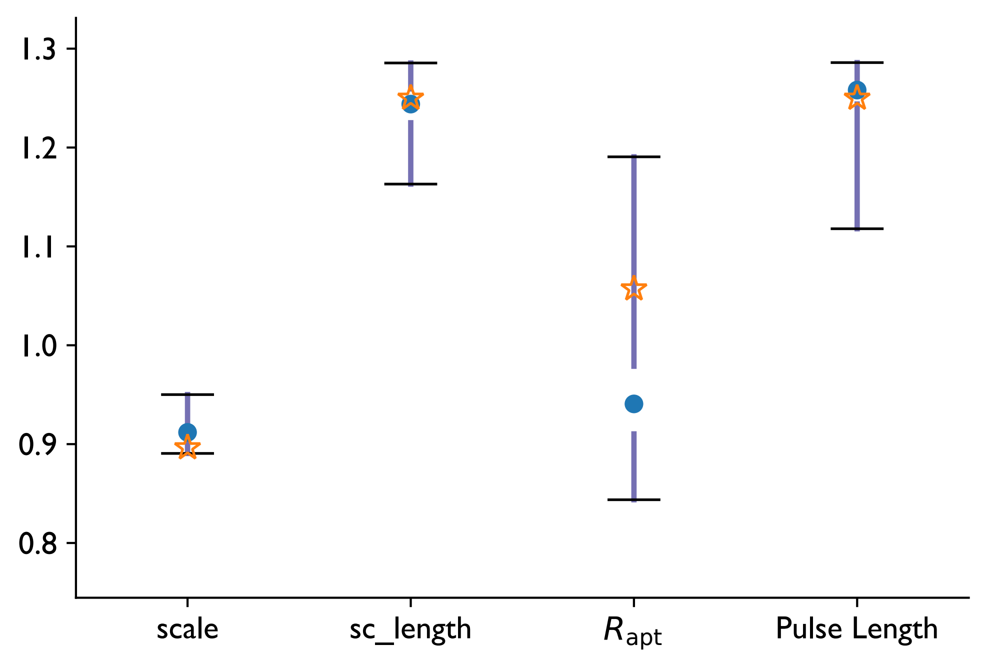
The inverse model attempts to map a Dante profile to the input parameters that produced it.

- A designer would like to work in the opposite direction:
  - Specify a desired temperature profile and have the model indicate what parameters would produce it.
- This could be done via optimization with the forward model: set an object and explore the design space.
- With neural networks (especially with dropout) we can try to directly learn this inverse map.
- One problem is that the map will not be unique: two sets of inputs can give the same (or nearly the same) output.
- Dropout can be used in the prediction to add noise (uncertainty) to the inverse models prediction.
  - Every evaluation takes a slightly different path through the network.
- The inverse network is “free” to evaluate, so we take 1000 evaluations and look at the distribution.
- The network has roughly the opposite structure to the forward network.



We test the inverse model with the output of a forward simulation.

- Using the forward model we used standard optimization to find a set of inputs that should give a flat profile at 0.275 keV.
- We then ran a new simulation using this hohlraum and gave the outputs to the inverse model.
- The inverse model results are shown in the figure.
- The error bars are the 95% confidence intervals of the 1000 evaluations.
    - The stars are the actual parameter values used in the simulation.
    - The dots are the medians of the inverse model evaluations.
- The R_{apt} parameter has a large uncertainty because it doesn't really matter.
- The other parameters are close to their true values.
- The inverse model gives a good starting point for a design study.



Using these models allows the optimization of the most important quantity, the uncertainty in the iron opacity.

- Given the theoretical behavior of the iron opacity (left), one can use different design parameters to minimize the uncertainty in the opacity measurement (Δσ).

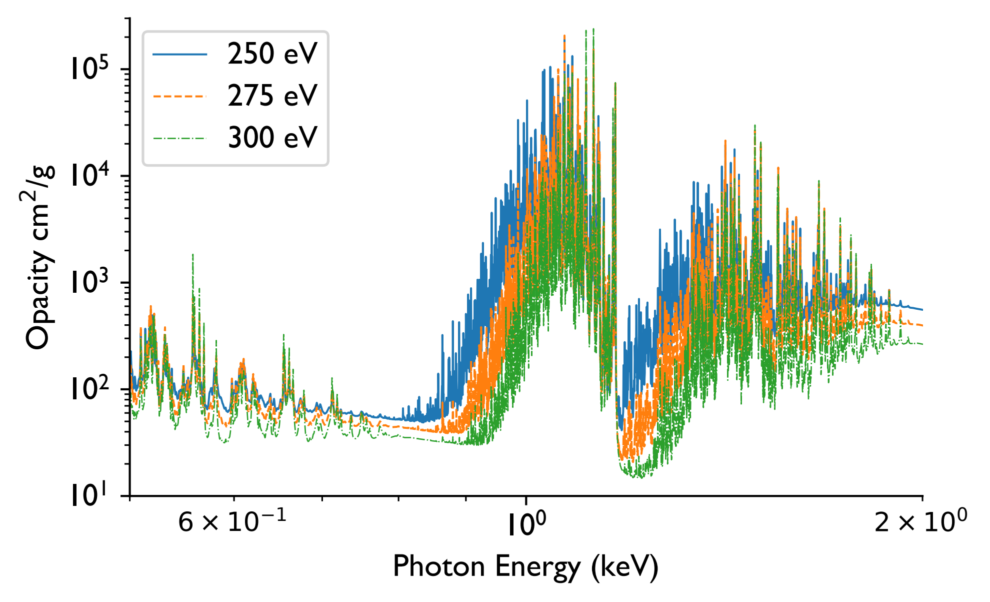


Figure 8: Iron opacity at three different temperatures and a density of 0.04 g/cm^3.

hohlraum	$\Delta\sigma$ (cm^2/g)		ΔE (eV)	
	near 800 eV	near 1200 eV	near 800 eV	near 1200 eV
sc_length= 0.8	8	15	29	2
sc_length= 1.25	2.4	3.9	24	22

There are many exciting challenges in HED Science.

- Many uncertainties that need to be dealt with.
- These problems require sophisticated computational science to solve.
- There are opportunities to apply machine learning to solve real problems.
- Data-driven and data-informed engineering are welcome to address some of these challenges.
- I also think that the techniques we have developed to solve these problems can be used in a variety of applications not just at these extreme conditions.

Thank you for your attention!