# Accelerated predictive modeling of the current profile evolution on NSTX-U using neural networks

Vaisnav Gajaraj[1,2], Mark Boyer[1], Michael Zarnstorff[1], Keith Erickson[1], Justin Kunimune[1,3]
[1]PPPL, [2]New York University, [3]Olin College of Engineering

**Abstract**
Fast, real time modeling of data will be vital for designing experiments and simulations for present day and future fusion devices like ITER. The modeling presented here focuses on the rapid evaluation of terms needed to evolve the magnetic diffusion equation and momentum diffusion equation for current and rotation profile prediction. A neural network has been developed to model plasma conductivity, bootstrap current, and flux surface averaged geometric quantities. The model drew from a database of 2016 NSTX-U TRANSP runs and used dimensionality reduction and an optimization algorithm to best select inputs, outputs, and hidden layer sizes. A fully connected neural network topology was used, and multiple models were trained to maximize profile prediction. Comparison of the models to test data shows that they can closely reproduce calculated profiles and scalar quantities relevant to the evolution of the magnetic diffusion equation. Combined with the recently developed NubeamNet model for beam current drive, these models demonstrate progress towards real time simulation of NSTX-U current profile evolution that can account for changes in plasma shaping.

## Introduction: TRANSP, diffusion equations and current profiles

Controlling the current and rotation profiles of plasmas in tokamak is imperative for the optimization of plasma experiments and for actively managing operations to prevent damage to the device. The computational models that calculate profiles across the plasmas must overcome large computational walls to model plasma profiles accurately, often at the cost of speed [8]. Often, the availability of computational resources and time constraints for particular experiments makes these physical models inappropriate for real-time control of plasmas. However, if there exists extensive databases of previously calculated computations using these models, neural network models can be created to reduce the operationally slow models into faster machine learned models. A fast and accurate profile model calculation from a neural networks will settle the timing compromises of large physics modeling codes.

Our project focused on reducing TRANSP's models of bootstrap current, resisivity, and flux surface averaged geometric quantities within the NSTX-U. TRANSP calculates time dependent analysis and diagnostic simulation of tokamak data [4]. With

NSTX-U data from the 2016 run, TRANSP solved the momentum  diffusion$^1$ and magnetic diffusion$^2$ equations to calculate models of profiles. The equation parameters rely on the following labeled profiles and input variables.

(1) $$nm\langle R^2\rangle\frac{\partial\omega}{\partial t}=\left(\frac{\partial V}{\partial\rho}\right)^{-1}\frac{\partial}{\partial\rho}\left[\frac{\partial V}{\partial\rho}nm\chi_\phi\langle R^2(\nabla\rho)^2\rangle\frac{\partial\omega}{\partial\rho}\right]+T_{\mathrm{NBI}}+T_{\mathrm{NTV}}.$$

$$\frac{\partial V}{\partial\rho}=DVOL$$

$$<R^2>=GR2$$

$$R^2(\nabla\rho)^2=GR2X2$$

$$\text{Inputs:}$$

$$\Sigma nm=NE$$

$$\chi_\phi=DIFB$$

(2) $$\frac{\partial\psi}{\partial t}=\frac{\eta(T_e)}{\mu_0\rho_b^2\hat{F}^2}\frac{1}{\hat\rho}\frac{\partial}{\partial\hat\rho}\left(\hat\rho\hat F\hat G\hat H\frac{\partial\psi}{\partial\hat\rho}\right)+R_0\hat H\eta(T_e)\frac{<\bar j_{NI}\cdot\bar B>}{B_{\phi,0}},$$

$$\hat\rho=\sqrt{}[\frac{TFLUX}{\pi B_0}]$$

$$\hat F=\frac{1}{GFUN}$$

$$\hat H=\frac{\hat F}{GR2I*R_0^2*100^2}$$

$$\hat G=100^4(R_0^2*GX2R2I*\rho_b^2)$$

$$\frac{<j_{BS}\cdot B>}{B_0}=\frac{CURBSSAU*GB2*GRI}{B_0^2*R_0*GFUN*GR2I}$$

Bootstrap profiles and flux surface averaged geometries both play a role in the evolution of these two equations that describe how magnetic fields diffuse in the plasma and how rotational energy diffuses in the plasma. Some terms, like $T_{NBI}$ and $T_{NTV}$ in the rotational diffusion equation$^1$, were not modeled in this work. Neutral beam injection ($T_{NBI}$) has its terms calculated by Dr. Dan Boyer's Neutral Beam Neural Net[1]. Future work requires the development of networks that calculate for the parameters of neoclassical toroidal viscosity $T_{NTV}$. The parameters solved for were the ones included in the breakdowns of the two diffusion equations as well as a few others that represented older bootstrap current models and other resistivity and flux measures.

For our bootstrap variable these include: CURBSESPS (Aspect Ratio Bootstrap Current), CURBSSAU (Sauter Bootstrap Current), CURBSWNC (Neoclassical

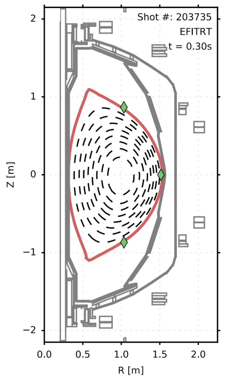
Bootstrap Current), CURGP (Grad-Shafranov Toroidal Current), ETA_SNC (Sauter Neoclassical Resistivity), ETA_SP (Spitzer Resistivity), ETA_SPS (Spitzer Resistivity Sauter), and ETA_WNC (Neoclassical Resistivity).

For the flux surface averaged geometries, the profiles being calculated for included, GFUN (Para/Diamagnetism), GX2R2I (Flux Surface Average), GR2I (Inverse Flux Surface Volume Average), Gr2 (Flux Surface Volume averages), GR2X2 (<1/R**2> Flux Surface Volume average), DVOL: Zone Volume, DAREA (Zone Cross Sectional Area), TFLUX (Enclosed Toroidal Flux). Among these parameters are several very similar flux surface measurements taken at different contour locations within the plasma.

When we mention bootstrap currents, we are referring to the particular profiles that describe a neoclassical toroidal current produced in the presence of a pressure gradient, neoclassical meaning that the currents are formed by complex particle orbits and drifts rather than classical colissional motion [5].

$$j_b \sim -\varepsilon^{1/2} \frac{1}{B_p} \frac{dp}{dr}$$

The bootstrap current {$j_b$}, where ε is a shaping constant in the tokamak, $B_p$ is the poloidal magnetic field, and p is plasma pressure [5] [6].



Flux surfaces in red and dashed black lines in the plotted diagram) are contours where the magnetic field vector does not cross, and TRANSP calculates the average of particular quantities along these contours for each time in an NSTX-U shot. In flux surfaces, equilibrium vector fields like the magnetic field B or current density j have simplified expressions, making these profile parameters important in the calculation of magnetic and momentum diffusion within the plasma [6].

## <u>Development of a Neural Network for Current Profile Prediction</u>

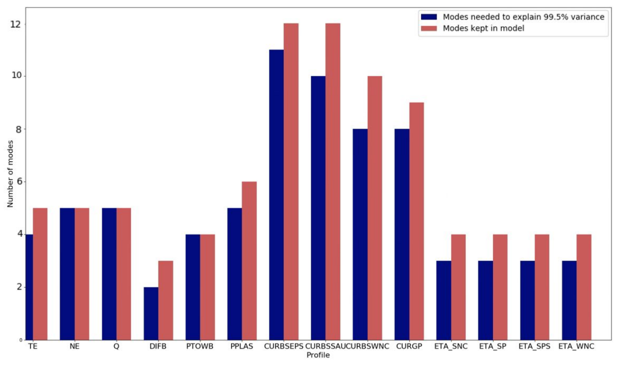
### Database Development

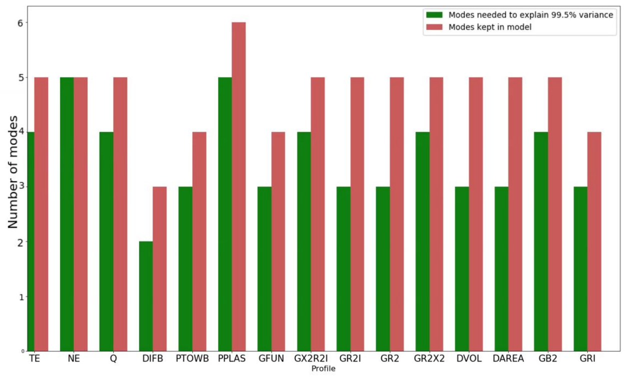
To quickly develop a reduced model framework that is both accurate and fast, we first had to make a limited database of TRANSP shots. To create a model for quick control applications, we found that it would be out of scope to gather a comprehensive dataset encompassing the complete physically possible range of all of the inputs to TRANSP, therefore we focused on a specific subset of inputs gathered from NSTX-U's first campaign in 2016. To generate the dataset, we did a randomized grid scan of 1000 NSTX-U shots with 5ms time steps and 10000 beam particles. For each shot being collected, a grid scan was defined and limited by key parameters being selected for including edge neutral density, shaping parameters, anomalous fast ion diffusivity, beam voltages, and axial plasma composition. The database contained nearly 100,000 time samples, of which eighty percent were randomly assigned to be used for the neural network training, ten percent were assigned to data validation, and the final ten percent were reserved for testing to determine the accuracy of our models [3]. The testing dataset was kept independent from the training models. The validation data was used to assess accuracy and generalization during hyper parameter tuning of the training set. The training set actively informed the weights of the neural network as it was being developed. Inputs to the model were chosen to be plasma pressure, kinetic MHD fast ion pressure, electron temperature, density profiles, safety (q) profile, and fast ion diffusivity.

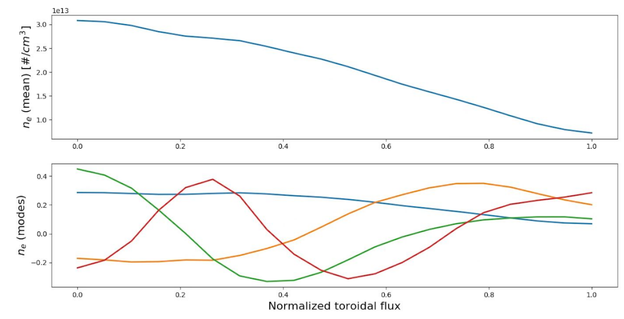
While training the data, we came across several issues with ability of our models to fit to our TRANSP training data, and had to add and subtract input profiles and scalars until we came across ones that provided enough information to calculate our desired output profiles. For example, we added the plasma pressure and the kinetic MHD fast ion pressure.

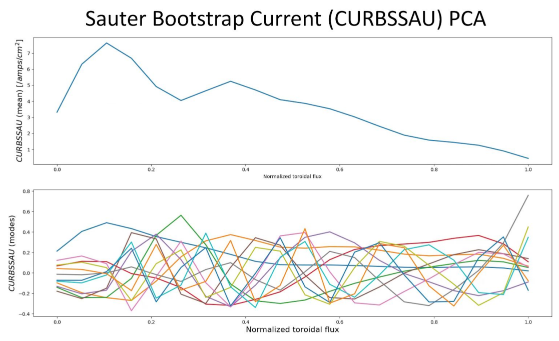
The outputs of our model were the aforementioned bootstrap current models and flux surface averaged geometries.

## Reducing Dimensionality of Profile Data





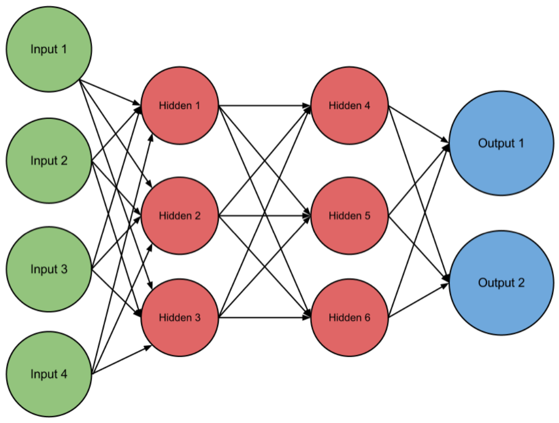




*Figures above represent the number of Principal component modes needed to explain 99.5% variance among the data. In blue are the bootstrap current profile modes and in green are the flux surface modes. Below each are the average principal component mode composition of a particular profile and the variety of other single modes compiled for the particular profile. In red are the actual number of modes used per profile in the final neural network, we add an extra mode for most profiles generally for good measure.*

To reduce the number of inputs and outputs being put into the neural network model we used a principal component analysis methodology to reduce the dimensionality of our inputs. Radially varying quantities are represented in TRANSP on a discrete grid of points in the normalized toroidal flux coordinate ρ, and these typically vary between 20 to 60 points. By applying principal component analysis to this data, we can reduce the size of the data and also the time it takes for the neural nets to compute results [3]. The radially varying quantities were projected onto a set of basis functions. The basis functions for each quantity were chosen by applying principal component analysis of the dataset and keeping only the most significant modes, typically between 4 and 10. It uses a covariance matrix to relate each profile variable to the others, then assigns directions with eigenvectors and assigns the relative importance of these directions with eigenvalues, essentially creating a framework of coefficient weights [12].

**Neural Networks**



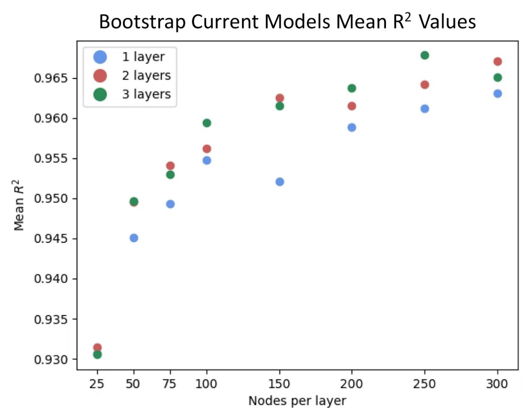
*Basic architecture of a neural network. Our networks varied up to 300 nodes per layer and up to 4 hidden layers.*

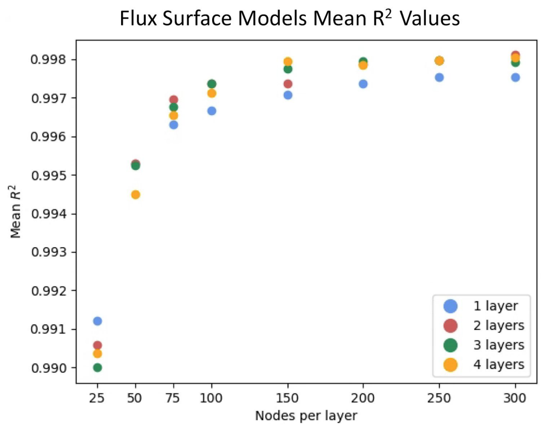
To develop our neural network we used the libraries provided by Sci-kit learn in python. These libraries allowed us perform PCA and hidden layer training. Neural Nets work by passing input data through a mesh of functions in hidden layers. Because the outputs desired were already available to us (from previous TRANSP runs) and we were trying to create a model to predict these outputs, our neural network was under the umbrella of supervised learning. Our supervised neural networks were developed by passing our selected input data through a weighted sum then passing it through a nonlinear 'activation' function [10]. In this way, a neural net creates a web like functional map from input to output. Specifically, vector projections of the training data onto the PCA assigns the 'weights' to the hidden layer transformations on the input data, which are essentially sensitivity coefficients for the inputs [12]. Multiple hidden layers act similarly to composite functions, increasing the sensitivity of layer outputs to the data. This may also cause overfitting of our data and the decrease of accuracy because the data is hyper sensitizing itself to noise [11]. The actual transformations are done in a black box regime, not allowing for much actual physical analysis. However for our control oriented purposes, this isn't hugely important.

## **<u>Model topology selection</u>**

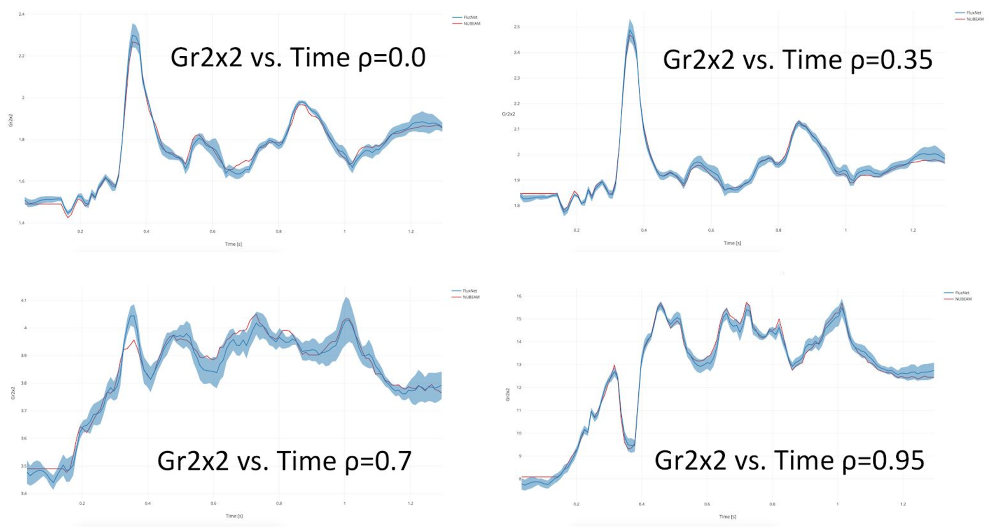
We trained multiple models of varying hidden layer amounts and node numbers and observed how well they fit to the TRANSP data. The choice of the number of hidden layers, hidden-layer nodes, and the regularization weight on the $L_2$-norm of the model coefficients was chosen through scoring how well models generalized to the shots in the validation dataset in a grid scan of hyperparameters. The standard deviation and range of the model predictions are used to provide estimates of the uncertainty of the predicted output. [3]. Looking at the regression lines of the models, we find that we get very strong matching between the neural network dataset and the validation set. We got up to an average of 99.8% matching for the Flux Surface averaged geometry profile values and up to 97% average regression matching for the Bootstrap current profiles.

Topology of models across several layers and node counts

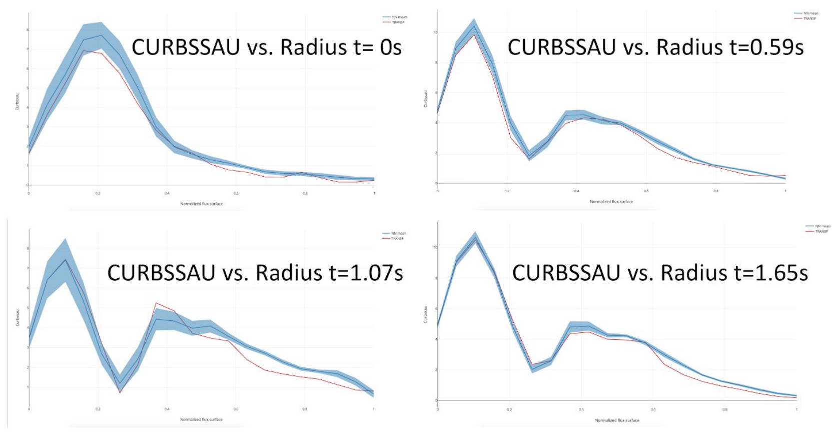




For the ones that exhibited the best regression fitting we were able to see good matching between the models and the training set.



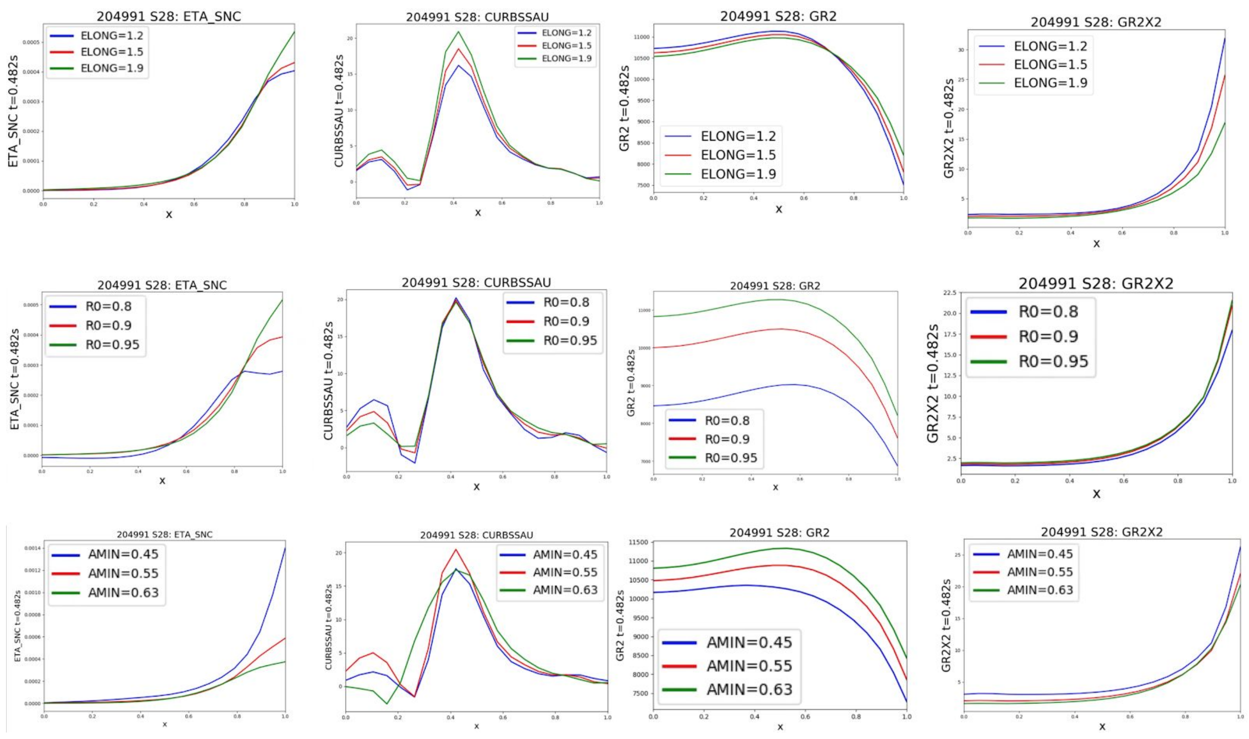
Time Traces of Flux Surface Average (GR2X2) for NSTX-U Shot #203878 against Neural Network of 3 layers with 100 nodes each



Radial Traces of Sauter Bootstrap Current (CURBSSAU) for NSTX-U Shot #204062 against Neural Network of 1 layer with 100 nodes

While these profiles are not perfect they do exhibit strong fitting that makes them suitable for quick control understandings and adjustments.

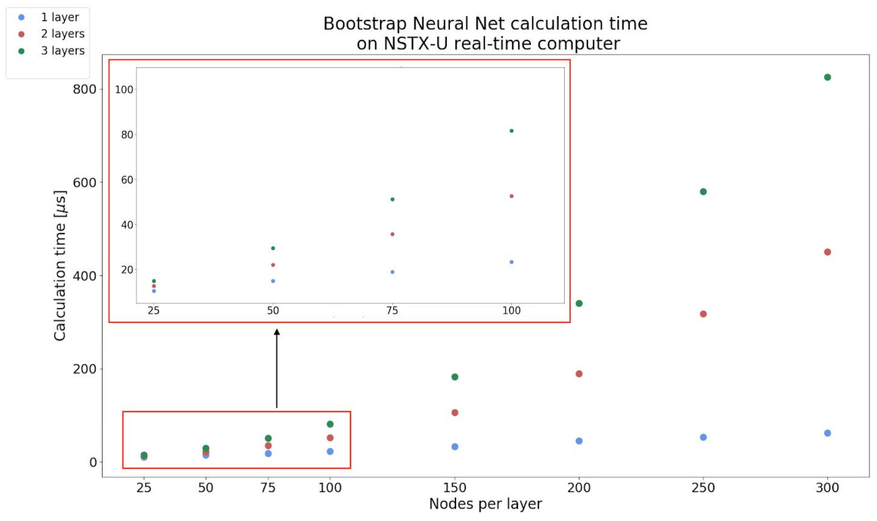
We were also successful in seeing the potential of our models to rapidly determine the effects of input changes across the set of outputs.



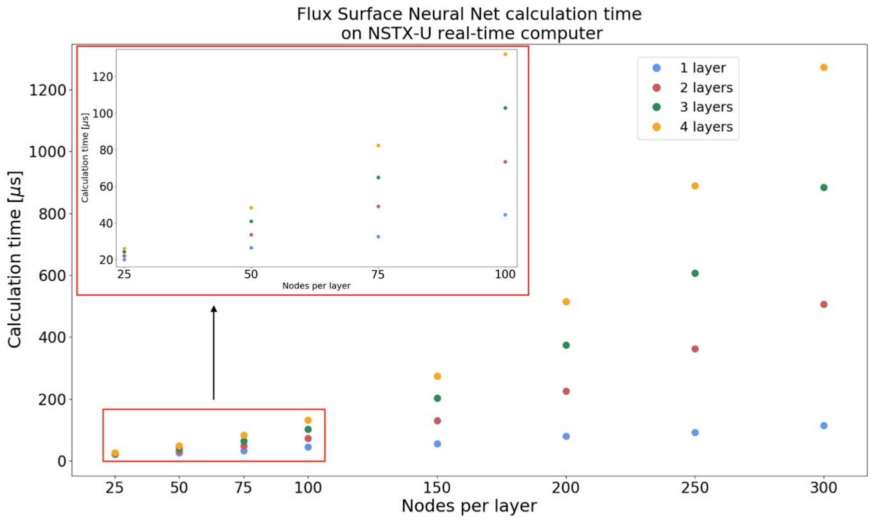
Using TRANSP's interpretive runs it takes too long to see the changes of these shaping parameters on the current profiles. However using the neural network model trained we can see the incremental changes in input scalar values like major radius (R0), flux surface elongation (ELONG), or minimum separatrix distance (AMIN), etc. on the current profiles. This opens the avenue for a new type of plasma control, one that focuses on the rapid management of the current profiles by changing the plasma's shape rather than adjusting neutral beam powers. Adjusting coils or the magnetic field to change the plasma's shape potentially adds more flexibility to control of the plasma profile.

## **Calculation Times**

Apart from accuracy, which was accomplished, fast calculation speeds were another goal of this control oriented approach to current profiles. To put the models to test, they were implemented in C++ and tested on a NSTX-U real-time computer (64 cores, 2.8Ghz, real-time kernel) to calculate processing speeds as if they were being used in a

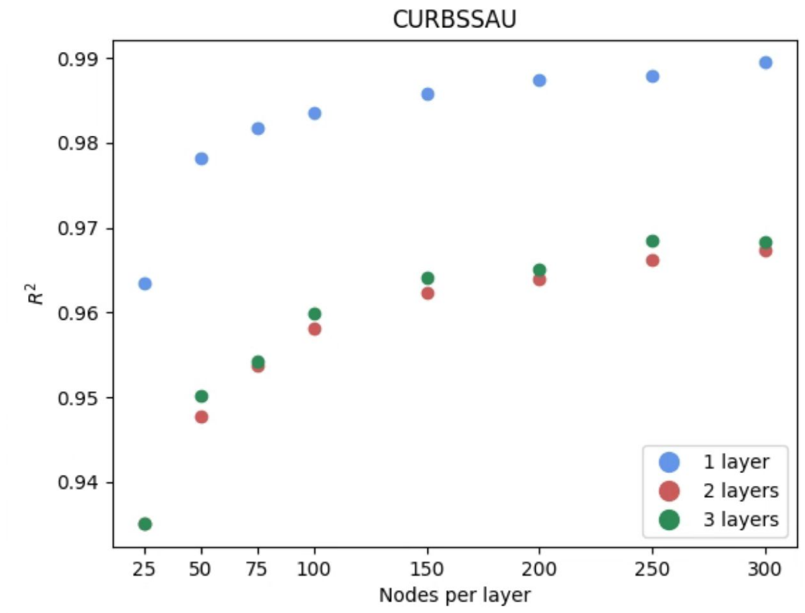


live experiment. Results successfully indicate that our profiles can be calculated in sub 130 microsecond time scales for the most significant and well fit models, proving the reliability of neural networks as a fast control alternative to TRANSP.



## Overfitting Phenomena

While developing our models we came across an interesting neural network phenomena that comes across when a dataset and the output it is trying to fit to is highly complex. As more layers are added, the model oversensitized the noise data, and fits for the noise thereby decreasing its accuracy. This is reflected in the regression plots' $R^2$ when the parameters are tested against the validation sets [11]. This was observed in the topologies of the sauter bootstrap current tested and other bootstrap profiles because of their complexity. This is a very well documented phenomena that must be reconciled with in the the development of a  neural network, i.e. the trade off between bias and variance. So while the model with only one hidden layer did not model the noise within the data, it was quite biased to the training data. Therefore, it might of fit with somewhat of a bias to the training data when measured against testing data. However, as seen in the testing data of CURBSSAU before, the data still fit fairly well when measured against testing data, meaning that even though there might of been bias to the training data, the neural network was still able to correctly fit testing data with about 3-5% error margins.



Example of overfitting of data due to the high variance within the profile data of sauter bootstrap current.

It should also be noted that while across all of the profile variables there tended to be an increase of validation fitting when the number of hidden layers were added, this is by no means a reflection of every profile. Each profile needs an evaluation of the correct number of nodes and hidden layers.

## **Future Work and Discussion**

We have successfully developed an array of neural networks for the purpose of accurate and fast current profile simulation and control in a spherical tokamak. Not only was our system accurate with less than 3% error margins when compared to testing data across all profiles, our models were able to compute in sub 100 microsecond timescales, a big step from the minute time scales a transp run might take to compute the same profiles.

Future work in this field might include repeating this data for other plasma profiles, such as the neoclassical toroidal viscosity term in the rotational diffusion equation. Trying to create similar models in a variety of splits of training, validation and testing data might also be of interest. Finally developing an accelerated solver of the magnetic and rotational diffusion equations with these neural networks would be useful in the regimes of plasma control as well.

For large scale analysis of these profiles using neural nets, the optimization of hidden layer and node selection is necessary. In this development, a rudimentary trial and error method that was used to pick nodes and hidden layers. Furthermore, our layers contained a homogeneous number of nodes through out. Whether or not there can be improvement in our models with different combinations of nodes is an open question that an optimization algorithm might be able to provide insight to.

## Reflection

I'd like to take this moment to thank Dr. Boyer, the Department of Energy, and everyone else who contributed to my summer at PPPL. This experience was hugely exciting and inspiring. Over the past 10 weeks, I've developed a strong drive to continue learning plasma physics and machine learning so that I might be able to contribute deeply to these fields myself one day.  I have learned so much, about physics, computer science and the value of research through this internship. I plan to continue learning more about the subjects discussed within this paper. I know that the skills I've gained at PPPL this summer will give me the foundation to pursue my goals as I progress in my career as a researcher.

## Bibliography

[1] Boyer, M. D., Barton, J., Shi, W., Wehner, W., Schuster, E., Ferron, J., . . . Penaflor, B. (2014). Simultaneous Boundary and Distributed Feedback Control of the Current Profile in H-mode Discharges on DIII-D. *IFAC Proceedings Volumes,47*(3), 1568-1573. doi:10.3182/20140824-6-za-1003.01712

[2] Goumiri, I., Rowley, C., Sabbagh, S., Gates, D., Gerhardt, S., Boyer, M., . . . Taira, K. (2016). Modeling and control of plasma rotation for NSTX using neoclassical toroidal viscosity and neutral beam injection. *Nuclear Fusion,56*(3), 036023. doi:10.1088/0029-5515/56/3/036023

[3] Boyer, M., Kaye, S., Liu, D., Erickson, K., Meneghini, O., & Sabbagh, S. (n.d.). Real-time capable Neural network approximation of NUBEAM for use in the NSTX-U control system.

[4] Hawryluk, R. (1981). An Empirical Approach To Tokamak Transport. *Physics of Plasmas Close to Thermonuclear Conditions,*19-46. doi:10.1016/b978-1-4832-8385-2.50009-1

[5] Miyamoto, K. (2005). *Plasma Physics and Controlled Nuclear Fusion*. New York: Springer-Verlag Berlin Heidelberg.

[6] Houlberg, W. A., Shaing, K. C., Hirshman, S. P., & Zarnstorff, M. C. (1997, September 01). Bootstrap current and neoclassical transport in tokamaks of arbitrary collisionality and aspect ratio. Retrieved from https://aip.scitation.org/doi/10.1063/1.872465

[7] TRANSP CODE FACT SHEET. (n.d.). Retrieved August 14, 2018, from https://w3.pppl.gov/topdac/transp.htm

[8] Meneghini, O., Smith, S., Snyder, P., Staebler, G., Candy, J., Belli, E., . . . Poli, F. (2017). Self-consistent core-pedestal transport simulations with neural network accelerated models. *Nuclear Fusion,57*(8), 086034. doi:10.1088/1741-4326/aa7776

[9] An introduction to machine learning with scikit-learn. (n.d.). Retrieved from http://scikit-learn.org/stable/tutorial/basic/tutorial.html#machine-learning-the-problem-setting

[10] LeCun, Y., Bengio, Y., & Hinton, G. (2015, May 27). Deep learning. Retrieved from https://www.nature.com/articles/nature14539

[11] Dietterich, T. (1995). Overfitting and undercomputing in machine learning. *ACM Computing Surveys,27*(3), 326-327. doi:10.1145/212094.212114

[12] Jolliffe, I. T., & Cadima, J. (2016). Principal component analysis: A review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences,374*(2065), 20150202. doi:10.1098/rsta.2015.0202