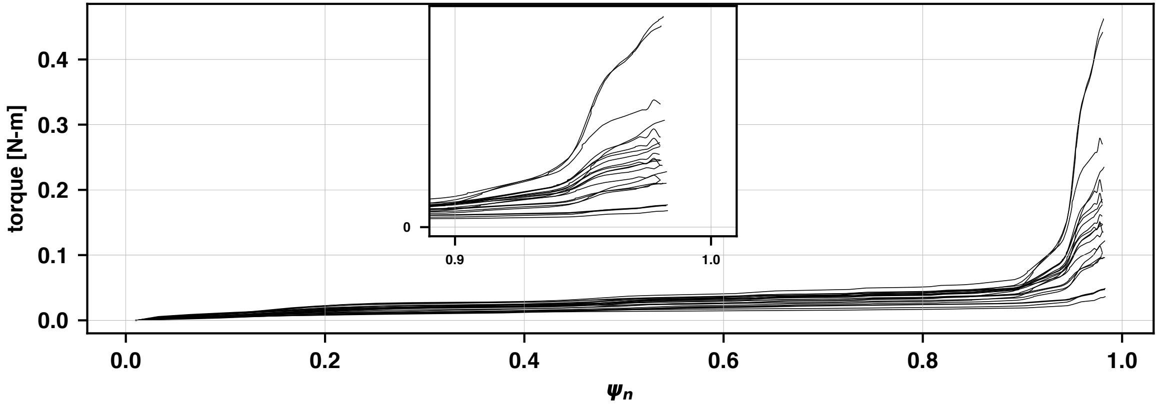
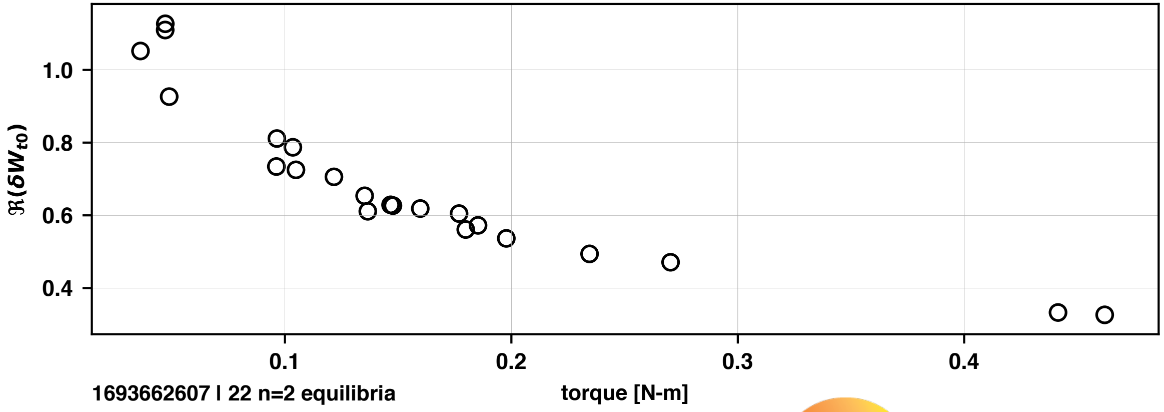
Neoclassical toroidal viscosity (NTV) torque prediction via deep learning

by
Mitchell Clement, Dan Boyer and Nik Logan

Presented at the NSTX-U/Magnetic fusion science meeting

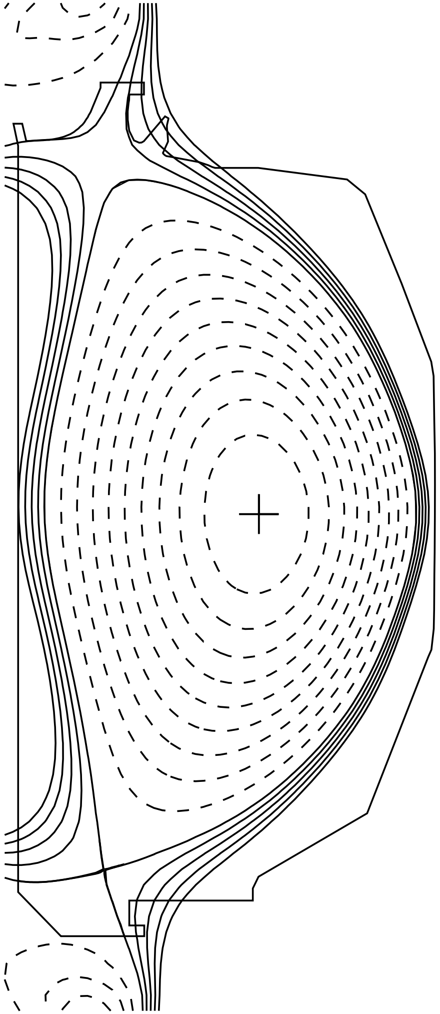
03/09/2020





Wide Pedestal Quiescent H-mode (WPQH) is a promising regime for burning plasmas in tokamaks

- **WPQH is a new stationary edge localized mode (ELM)-free regime obtained at DIII-D**
- **Balanced co and counter beam injection often results in low rotation**
- **E_r and ExB shear are integral to pedestal performance**
  - Rotation profile shape has a large affect on these quantities
- **A perfect scenario for <u>rotation profile control</u>**



Motivation

- **Toroidal momentum equation describes rotation profile evolution in toroidal plasma**

$$\sum_i n_i m_i \langle R^2\rangle \frac{\partial \omega}{\partial t} + \omega \langle R^2\rangle \sum_i m_i \frac{\partial n_i}{\partial t} + \sum_i n_i m_i \omega \frac{\partial \langle R^2\rangle}{\partial t} + \sum_i n_i m_i \langle R^2\rangle \omega \left(\frac{\partial V}{\partial \rho}\right)^{-1} \frac{\partial}{\partial t}\frac{\partial V}{\partial \rho} =$$

$$\left(\frac{\partial V}{\partial \rho}\right)^{-1} \frac{\partial}{\partial \rho}\left[\frac{\partial V}{\partial \rho}\sum_i n_i m_i \chi_\varphi \langle R^2 (\nabla\rho)^2\rangle \frac{\partial \omega}{\partial \rho}\right] - \left(\frac{\partial V}{\partial \rho}\right)^{-1} \frac{\partial}{\partial \rho}\left[\frac{\partial V}{\partial \rho}\sum_i n_i m_i \omega \langle R^2 (\nabla\rho)^2\rangle \frac{v_\rho}{|\nabla\rho|}\right] - \sum_i n_i m_i \langle R^2\rangle \omega \left(\frac{1}{\tau_{\varphi cx}} + \frac{1}{\tau_{c\delta}}\right) + \sum_i \frac{\partial T_j}{\partial V}$$

$$\left.\frac{\partial \omega}{\partial \rho}\right|_{\rho=0} = 0$$

$$\omega|_{\rho=1} = 0$$

DIII-D NATIONAL FUSION FACILITY

Motivation

- **Toroidal momentum equation describes rotation profile evolution in toroidal plasma**
- **Simplifying assumptions reduce PDE complexity**

$$nm\langle R^2\rangle \frac{\partial \omega}{\partial t} = \left(\frac{\partial V}{\partial \rho}\right)^{-1} \frac{\partial}{\partial \rho}\left[\frac{\partial V}{\partial \rho}\sum_i n_i m_i \chi_\varphi \langle R^2 (\nabla \rho)^2\rangle \frac{\partial \omega}{\partial \rho}\right] + \frac{\partial T_{NBI}}{\partial V} + \frac{\partial T_{NTV}}{\partial V} + \frac{\partial T_{int}}{\partial V}$$

$$\left.\frac{\partial \omega}{\partial \rho}\right|_{\rho=0} = 0$$

$$\left.\omega\right|_{\rho=1} = 0$$

Motivation

- **Toroidal momentum equation describes rotation profile evolution in toroidal plasma**
- **Simplifying assumptions reduce PDE complexity**

$$nm\langle R^2\rangle \frac{\partial \omega}{\partial t} = \left(\frac{\partial V}{\partial \rho}\right)^{-1} \frac{\partial}{\partial \rho}\left[\frac{\partial V}{\partial \rho}\sum_i n_i m_i \chi_\varphi \langle R^2 (\nabla \rho)^2\rangle \frac{\partial \omega}{\partial \rho}\right] + \frac{\partial T_{NBI}}{\partial V} + \frac{\partial T_{NTV}}{\partial V} + \frac{\partial T_{int}}{\partial V}$$

$$\left.\frac{\partial \omega}{\partial \rho}\right|_{\rho=0} = 0$$

$$\omega|_{\rho=1} = 0$$

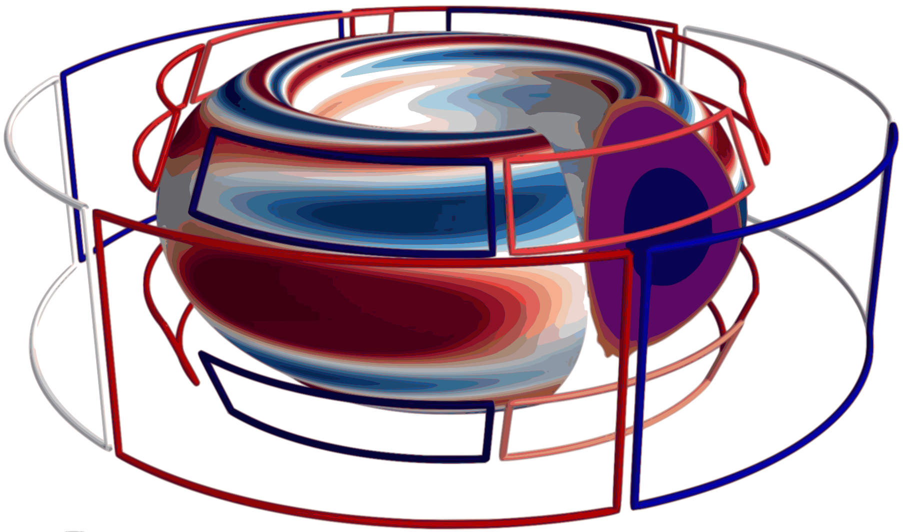
This talk will cover

- **Wide Pedestal Quiescent H-mode (WPQH)**
- **What is neoclassical toroidal viscosity (NTV) torque?**
  - General Perturbed Equilibrium codes (GPEC)
- **GPECNet**
  - A neural network model to predict NTV torque
- **Conclusions and next steps**

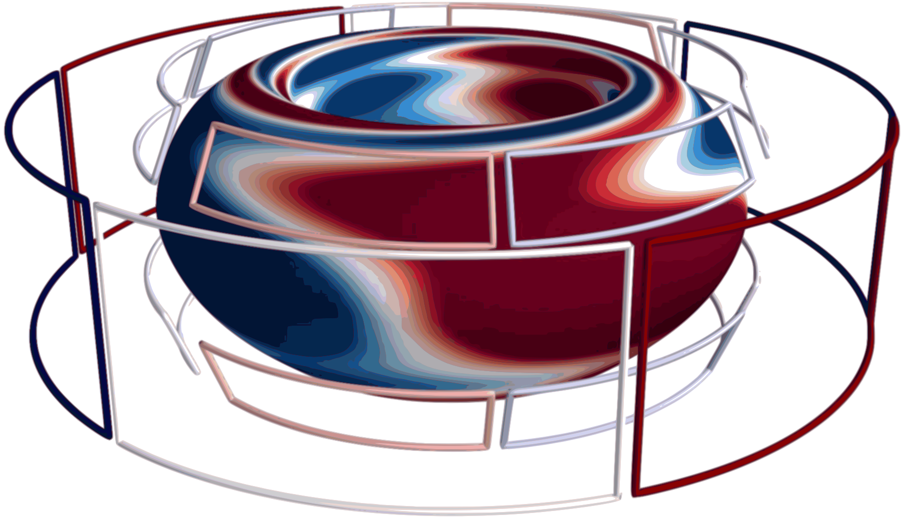
Neoclassical toroidal viscosity (NTV) Torque

- **Torque driven by single n (n = 1-3) 3D fields ($\delta B_n/B_0 \geq 10^{-3}$)**
- **i.e. <u>3D coils can have a braking effect on plasma rotation</u>**

NTV in the Edge



NTV Throughout



N. Logan, APS 2018

Previous work in rotation profile control used simplified model for NTV

- **Previous work in rotation profile control used a simple Gaussian model for NTV torque:**

$$\frac{\partial T_{NTV}}{\partial V}(t,\rho) = -KG(\rho)\langle R^2\rangle I^2(t)\omega(t,\rho)$$

K = const.

G = Gaussian function

ω = rotation profile

I = coil current

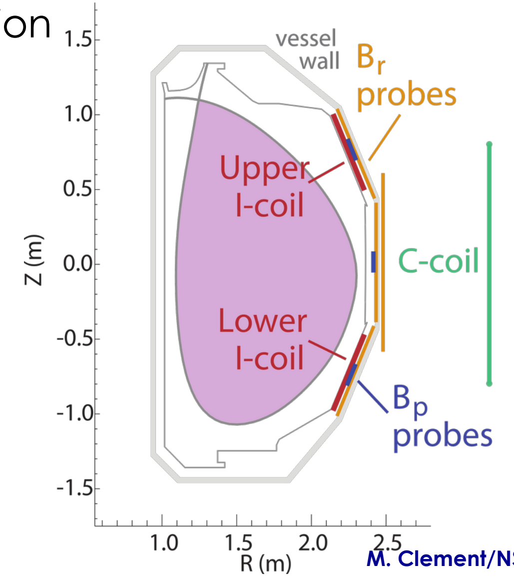
- **Controller used Linear Quadratic Integral control**

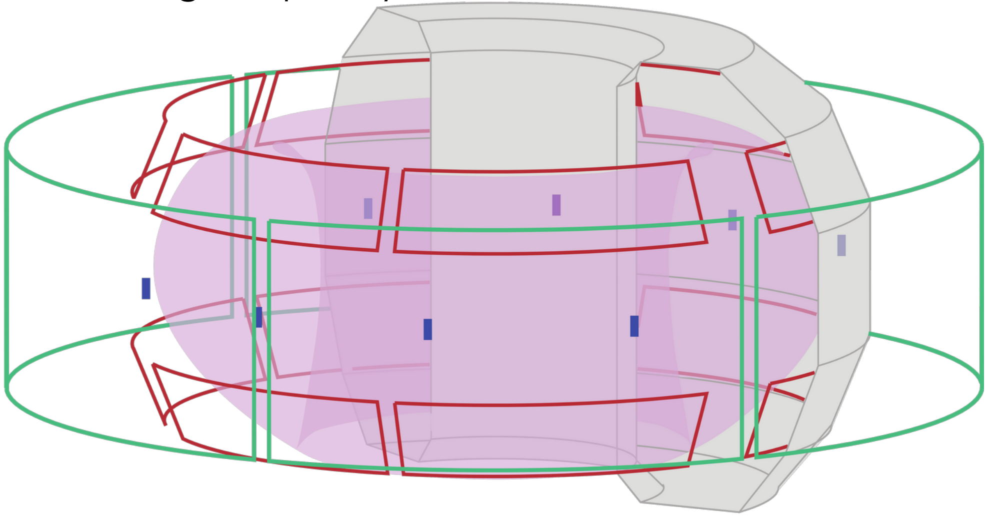
DIII-D NATIONAL FUSION FACILITY

I. R. Goumiri et. Al (2016) Nuclear Fusion, 56(3), 036023

General Perturbed Equilibrium Codes (GPEC) includes Direct Criterion of Newcomb (DCON)

- **DCON takes an equilibrium (EFIT) and toroidal mode number (n=2)**
  - Does MHD δW analysis, outputs δWs for given n
- **GPEC takes DCON output files, kinetic profiles of $n_e(\psi_n)$, $n_i(\psi_n)$, $T_e(\psi_n)$, $T_i(\psi_n)$, $\Omega_{ExB}(\psi_n)$ and 3D coil configuration (n=2)**
  - Outputs many MHD related quantities, including torque by coil matrix for coil n configuration





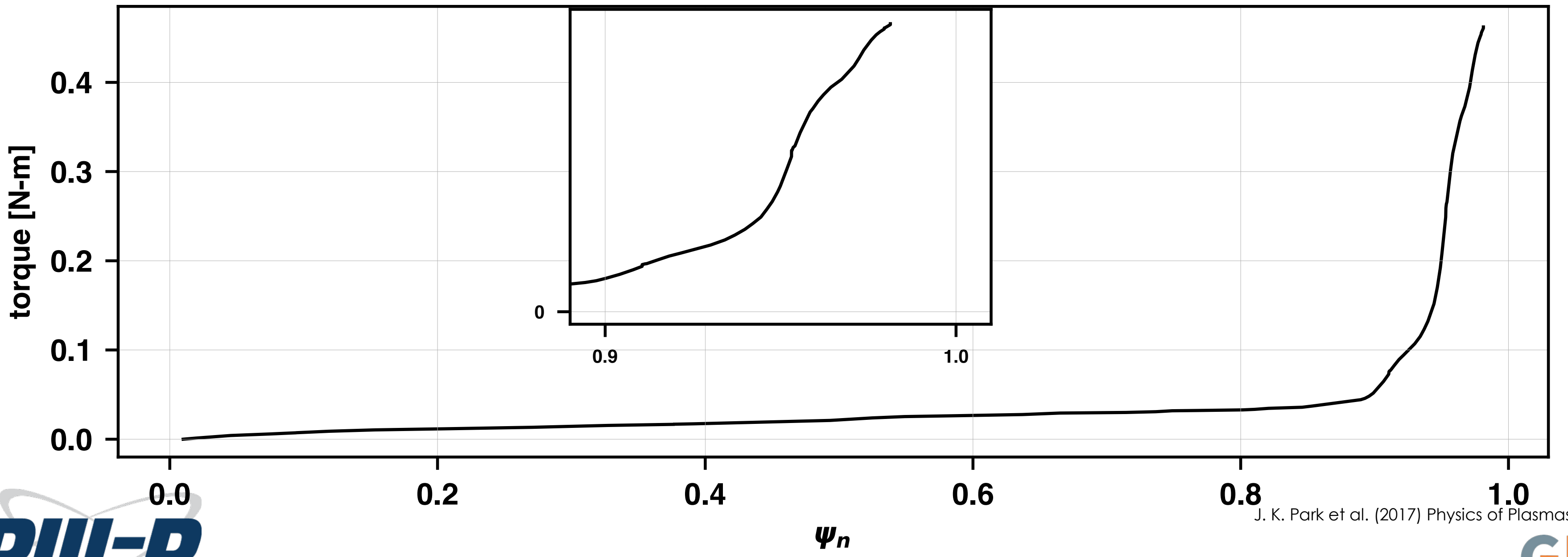
Glasser, A. H. (2016) Physics of Plasmas, 23(7), 072505

General Perturbed Equilibrium Codes (GPEC) calculated torque matrix profile

- **GPEC calculates torque by coil matrix as a function of ψ, NTV torque is expressed as**

$$T_{NTV}(\psi) = \vec{I}^{T}\mathbf{T}_{coil}(\psi)\vec{I}$$

where $\vec{I}^{T} = \begin{bmatrix} I_C^{IU} & I_S^{IU} & I_C^{IL} & I_S^{IL} & I_C^{C} & I_S^{C} \end{bmatrix}$



J. K. Park et al. (2017) Physics of Plasmas, 24(3), 032505

General Perturbed Equilibrium Codes (GPEC) calculated torque matrix profile

- **Torque density by coil matrix is not directly calculated by GPEC, but can be computed from other quantities:**

$$\frac{\partial \mathbf{T}_{coil}}{\partial V}(\psi) = \frac{\partial \mathbf{T}_{coil}}{\partial \psi}(\psi)\frac{\partial \psi}{\partial V}(\psi)$$

$$\frac{\partial T_{NTV}}{\partial V}(\psi) = \vec{I}^{T}\frac{\partial \mathbf{T}_{coil}}{\partial V}(\psi)\vec{I}$$

- **$T_{coil}(\psi)$ is a 3D tensor (profile of matrices), interpolated to 200x6x6 array**

DIII-D NATIONAL FUSION FACILITY

GPEC
http://princetonuniversity.github.io/GPEC/

Can a deep learning model be derived from GPEC results for real-time NTV profile prediction?

- **GPEC is not a real-time code**
  - GPEC can take anywhere from 8 mins to over an hour to compute results (parallelization dependent)
- **Neural networks trained to predict neutral beam torque profiles have shown promising results**
  - NUBEAMnet trained on TRANSP/NUBEAM results

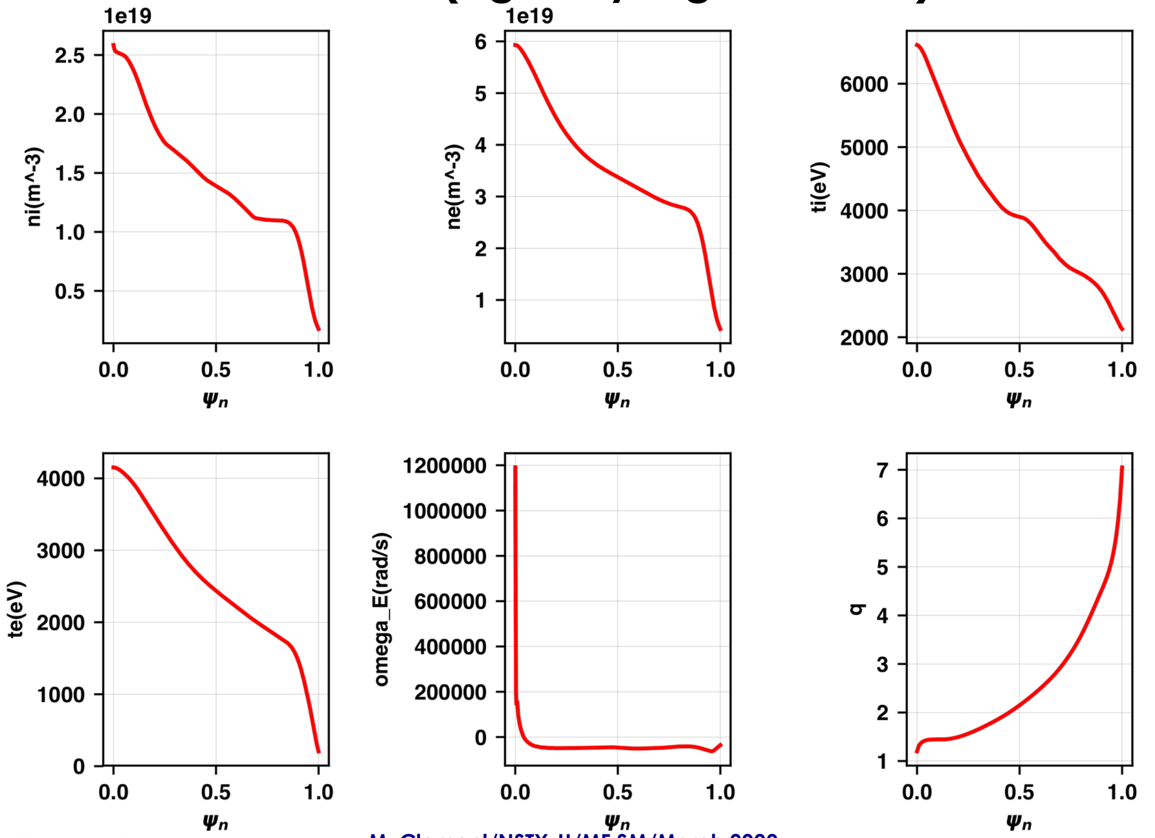
M.D. Boyer et al. (2019) Nuclear Fusion, 59(5), 056008

Compiled database of 2468 DCON/GPEC runs from existing results

- **Found every WPQH TRANSP run that already existed**
  - Only ~294 WPQH shots (multiple runids for a single shot)
  - Each runid sampled every ~100ms, yielded 2291 equilibria
  - TRANSP equilibrium (gEQDSK) and kinetic profiles used as inputs for DCON and GPEC runs
  - n=2 for all runs
  - No control over quality of runs
- **Scraped T. Osborne's SQL database for existing profiles**
  - Yielded 177 equilibria
  - EFITs and profiles used as inputs for DCON and GPEC runs
  - Profiles used for 50ms TRANSP runs
  - n=2 for all runs
  - No control over quality of profiles
- **2468 total equilibria**

Database filtered for unrealistic inputs and outputs

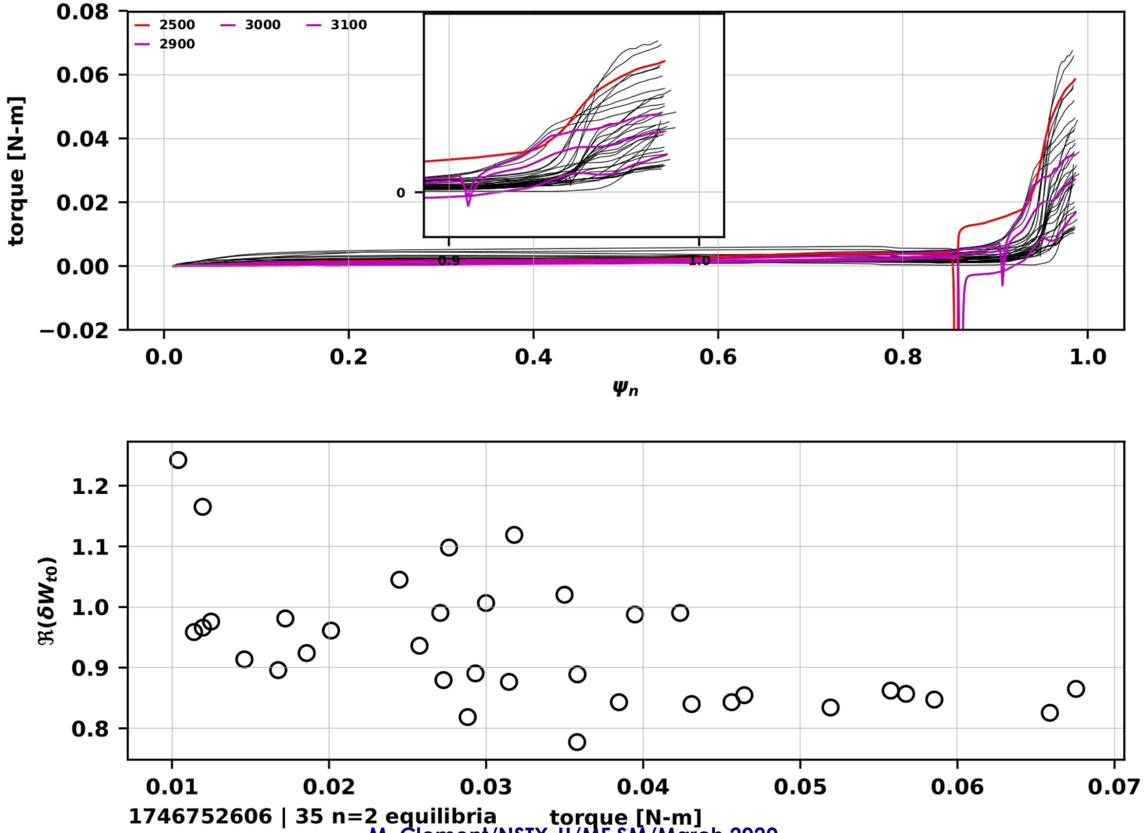
- **Some input profiles are unrealistic (eg. very high rotation)**



1635182602.2700

Database filtered for unrealistic inputs and outputs

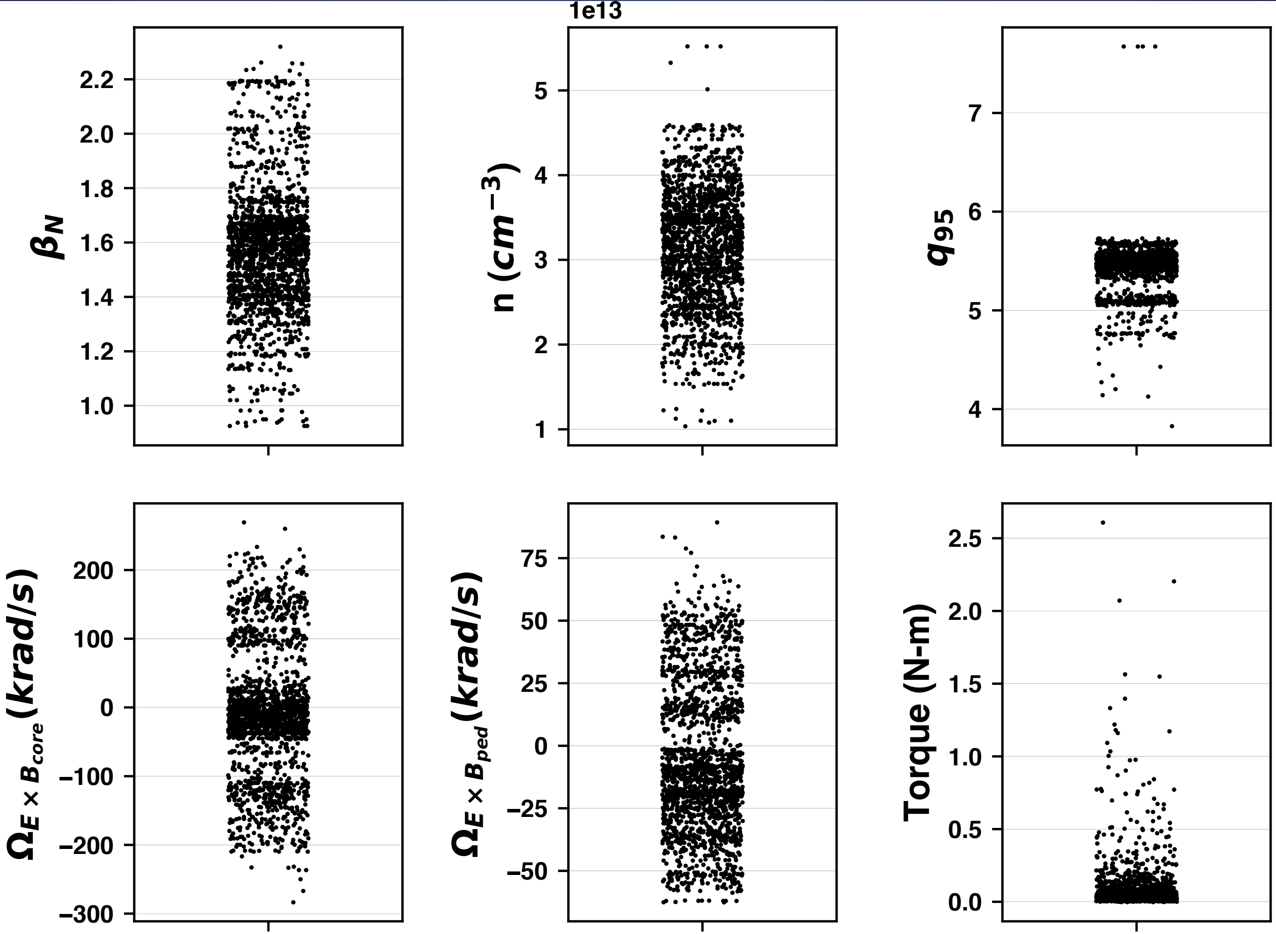
- Some output torque profiles are unrealistic



M. Clement/NSTX-U/MF SM/March 2020

DIII-D NATIONAL FUSION FACILITY

Database represents wide operating space of WPQH parameters (1856 usable equilibria)

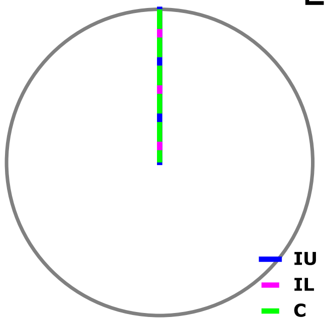


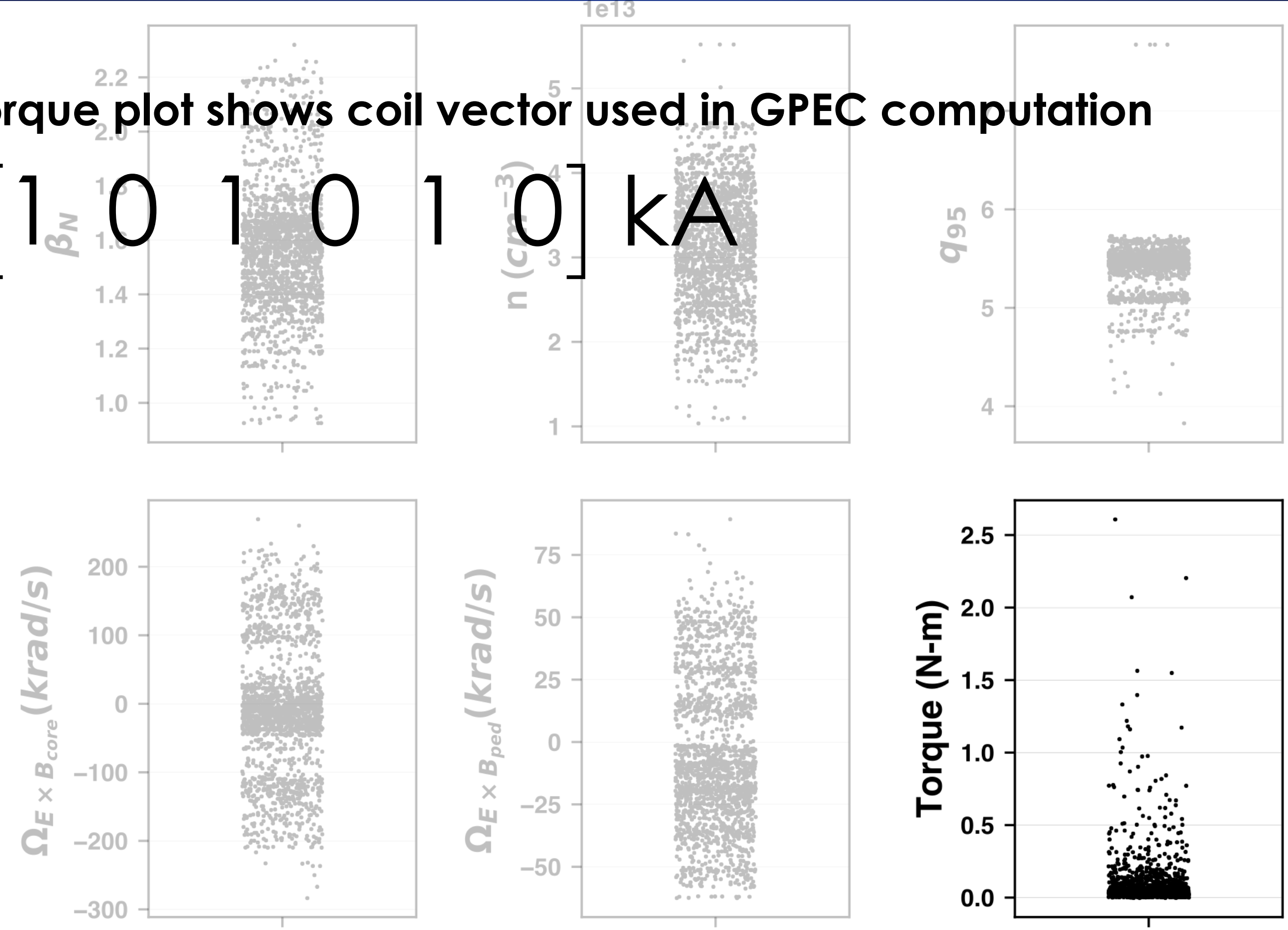
GPECnet database | 2468 equilibria | 612 blacklisted | 1856 whitelisted

Database represents wide operating space of WPQH parameters (1856 usable equilibria)

- **Integral torque plot shows coil vector used in GPEC computation**

$$\vec{I}^T = [1\ 0\ 1\ 0\ 1\ 0]\ \mathrm{kA}$$





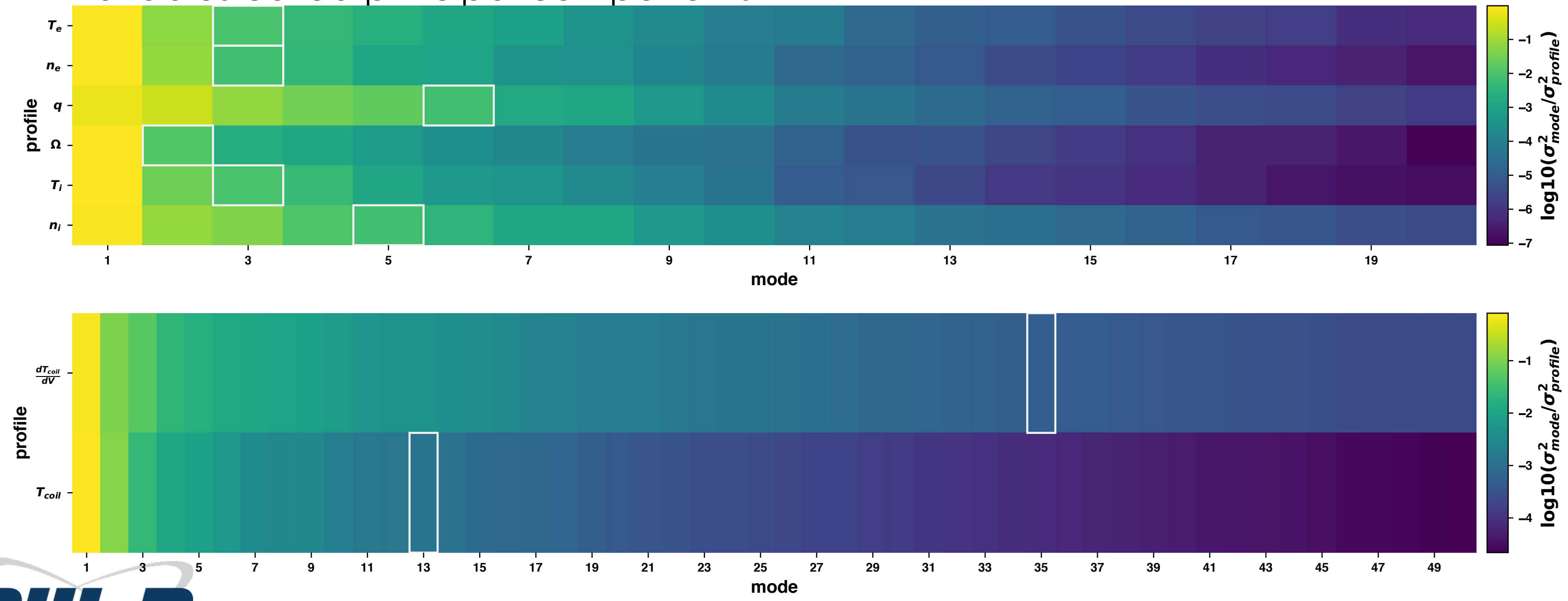
GPECnet database I 2468 equilibria I 612 blacklisted I 1856 whitelisted

Database used to train neural network to predict δW_0, T_{coil} and dT_{coil}/dV

- **Built with Keras python module (TensorFlow backend)**
- **Neural Network inputs (from TRANSP):**
  - Scalars – effective charge (**Z_{eff}**), major radius (**R_0**), elongation (**κ**), plasma current (**I_p**), minor radius (**a**), vacuum toroidal field (**$B_{\varphi,v}R$**), upper triangularity (**δ_u**), lower triangularity (**δ_l**)
  - Profiles – electron temperature (**T_e**), ion temperature (**T_i**), electron density(**n_e**), safety factor (**q**), rotation (**Ω**) (reduced via Principle Component Analysis [PCA])
- **Neural Network outputs (in separate models):**
  - 1st or least stable mode from DCON (**δW_0**)
  - Torque by coil or Torque density by coil matrix profile (**T_{coil}, $dT_{coil}/dV(\rho)$**)

Principle Component Analysis (PCA) reduces dimension of profile data via SVD

- **PCA:**
  - statistical procedure that uses an orthogonal transformation to convert a set of observations of possibly correlated variables into a set of values of linearly uncorrelated variables called principal components



DIII-D NATIONAL FUSION FACILITY

GPECNet results are predictions of unseen test set data

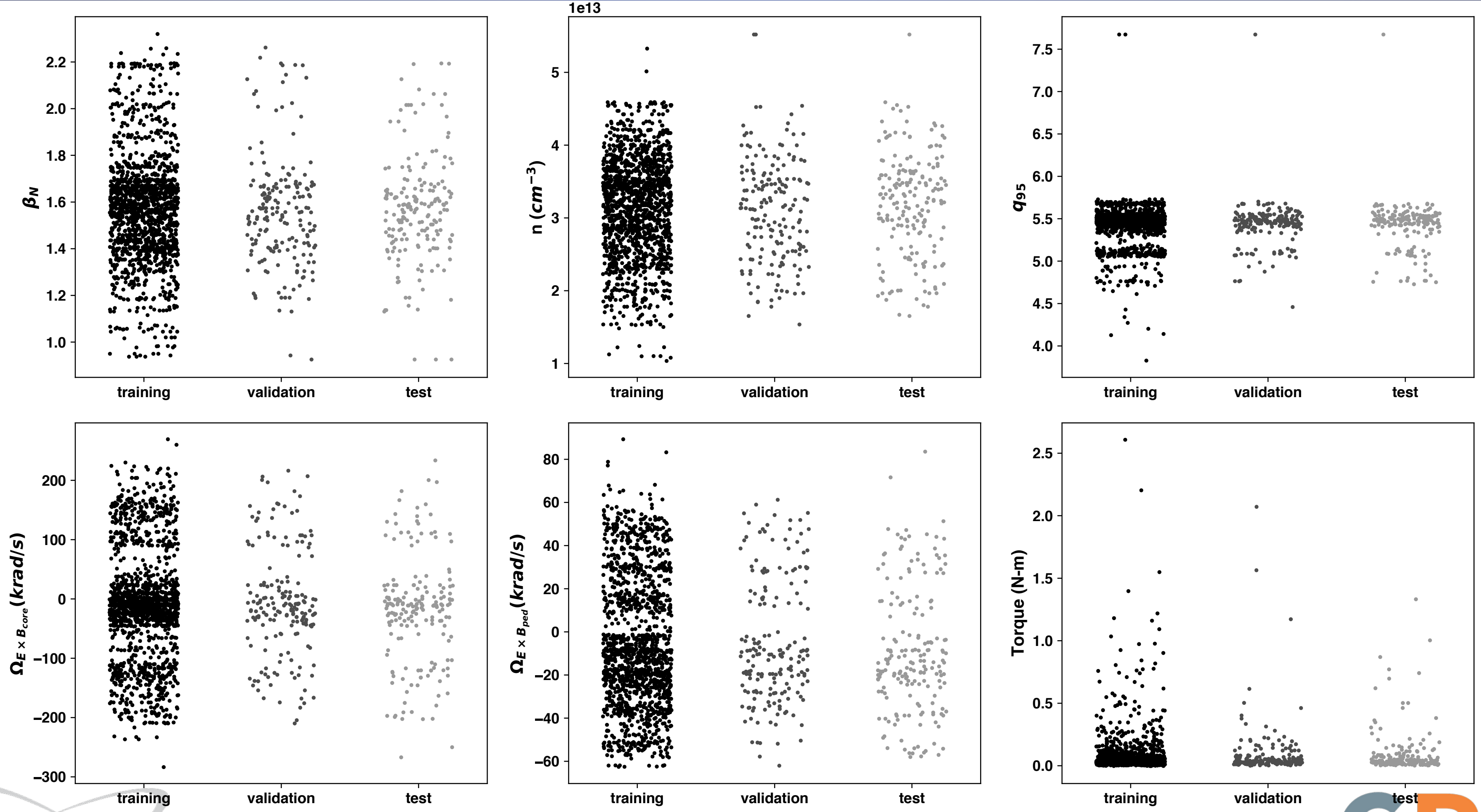
- **Database breakdown:**
  - 80% (1484) for training
  - 10% (185) for validation (used for gradient descent)
  - 10% (187) for testing (aka “test set”)
  - Equilibria selected at random for each set
- **GPECnet never “sees” test set data during the training process**
  - Therefore, predictions and comparisons shown are made using test set data
- **Random coil current vector used to evaluate T_{NTV} and dT_{NTV}/dV**
  - Generate 3 random phase angles: θ_{IU}, θ_{IL}, θ_{C}

$$\vec{I}^T = \left[\cos(n\cdot\theta_{IU})\ \ \sin(n\cdot\theta_{IU})\ \ \cos(n\cdot\theta_{IL})\ \ \sin(n\cdot\theta_{IL})\ \ \cos(n\cdot\theta_{C})\ \ \sin(n\cdot\theta_{C})\right]\ \mathrm{kA}$$

$$T_{NTV}(\rho) = \vec{I}^T \mathbf{T}_{coil}(\rho)\vec{I}$$

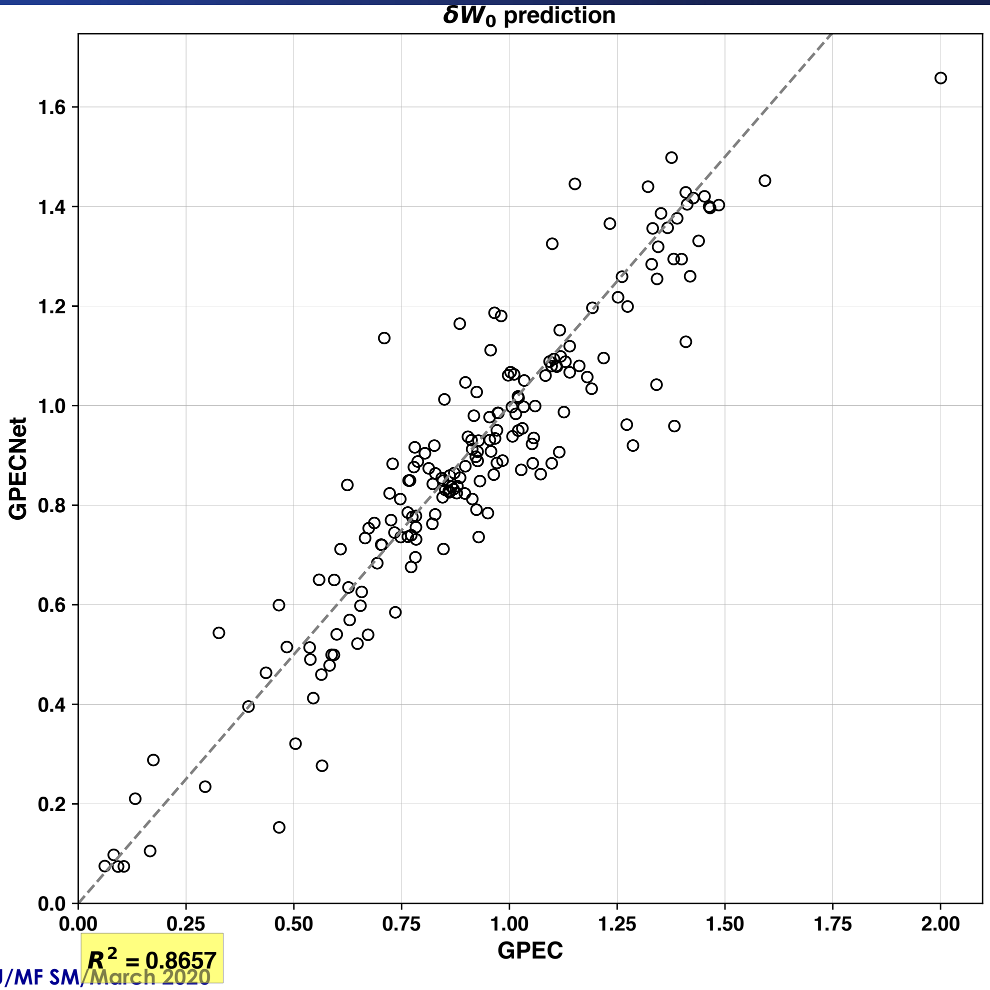
$$\frac{\partial T_{NTV}}{\partial V}(\rho) = \vec{I}^T \frac{\partial \mathbf{T}_{coil}}{\partial V}(\rho)\vec{I}$$

GPECNet results are predictions of unforeseen test set data



GPECNet δW_0 scalar regression is straightforward to interpret

- **δW_0 model:**
  - 3 Dense hidden layers with 100, 50 and 33 nodes ("relu" activation); 1 Dense output layer ("linear" activation)
- **Coefficient of determination, R^2, determines overall quality of predictions**



GPECNet profile regression is much less straightforward

- Neural Net must predict a profile of matrices (T_{coil} or dT_{coil}/dV)
- Root Mean Squared Errors (RMSE) used to measure performance

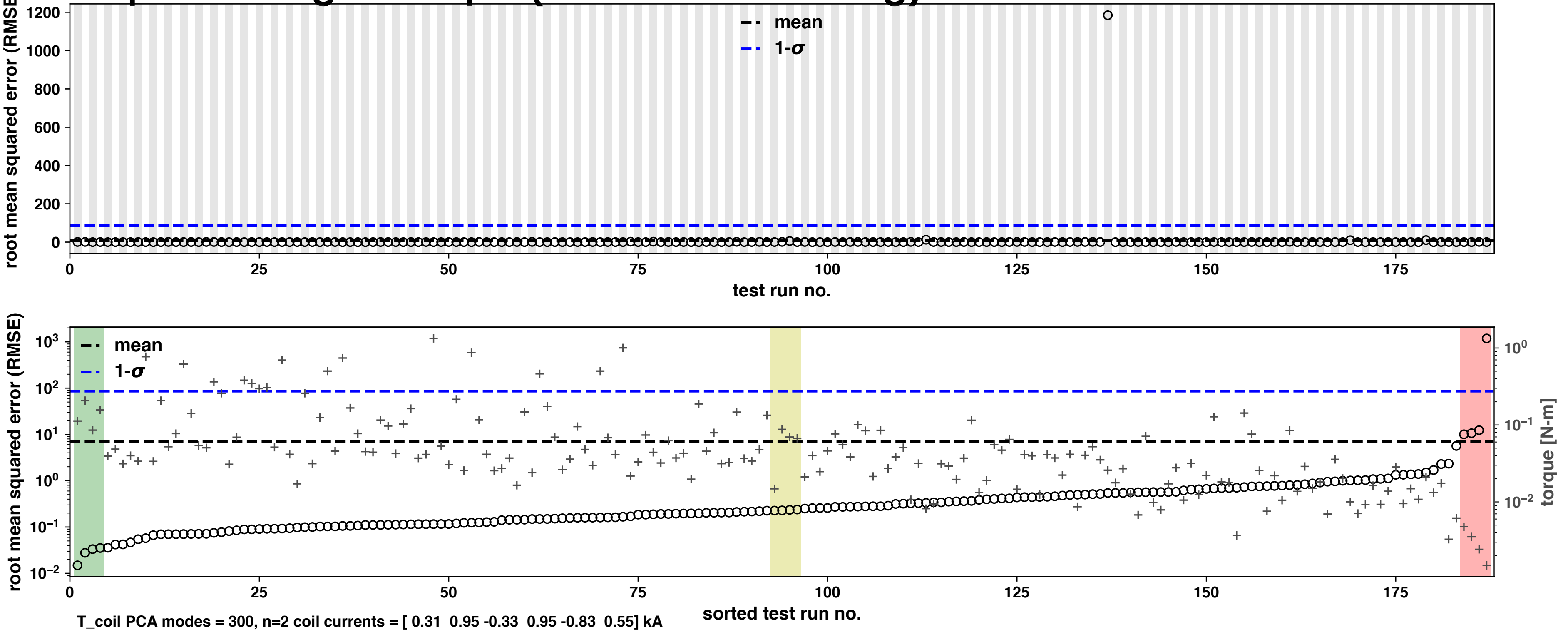
$$RMSE_{\frac{\partial T}{\partial V}} = \sqrt{\frac{1}{N}\sum_{i=0}^{N}\left(\frac{\frac{\partial T_i}{\partial V} - \frac{\partial \hat{T}_i}{\partial V}}{\left|\frac{\partial T}{\partial V}\right|_{max}}\right)^2}$$

$$RMSE_T = \sqrt{\frac{1}{N}\sum_{i=0}^{N}\left(\frac{T_i - \hat{T}_i}{T_{\rho=1}}\right)^2}$$

DIII-D NATIONAL FUSION FACILITY

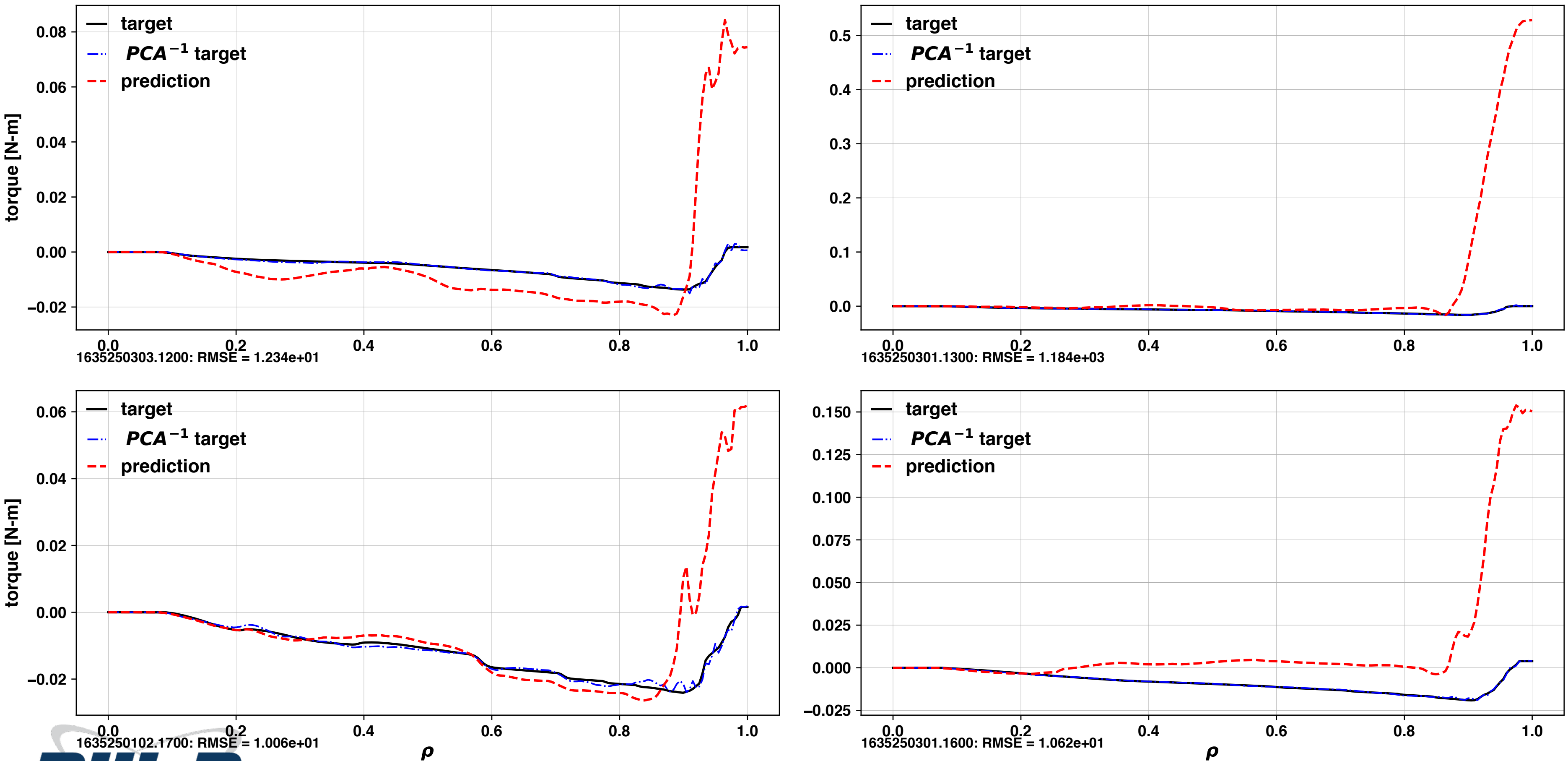
GPECNet T_{coil} profile regression uses RMSE to gauge model performance

- NN predicts higher torque (GPEC run coil config) cases with less RMSE

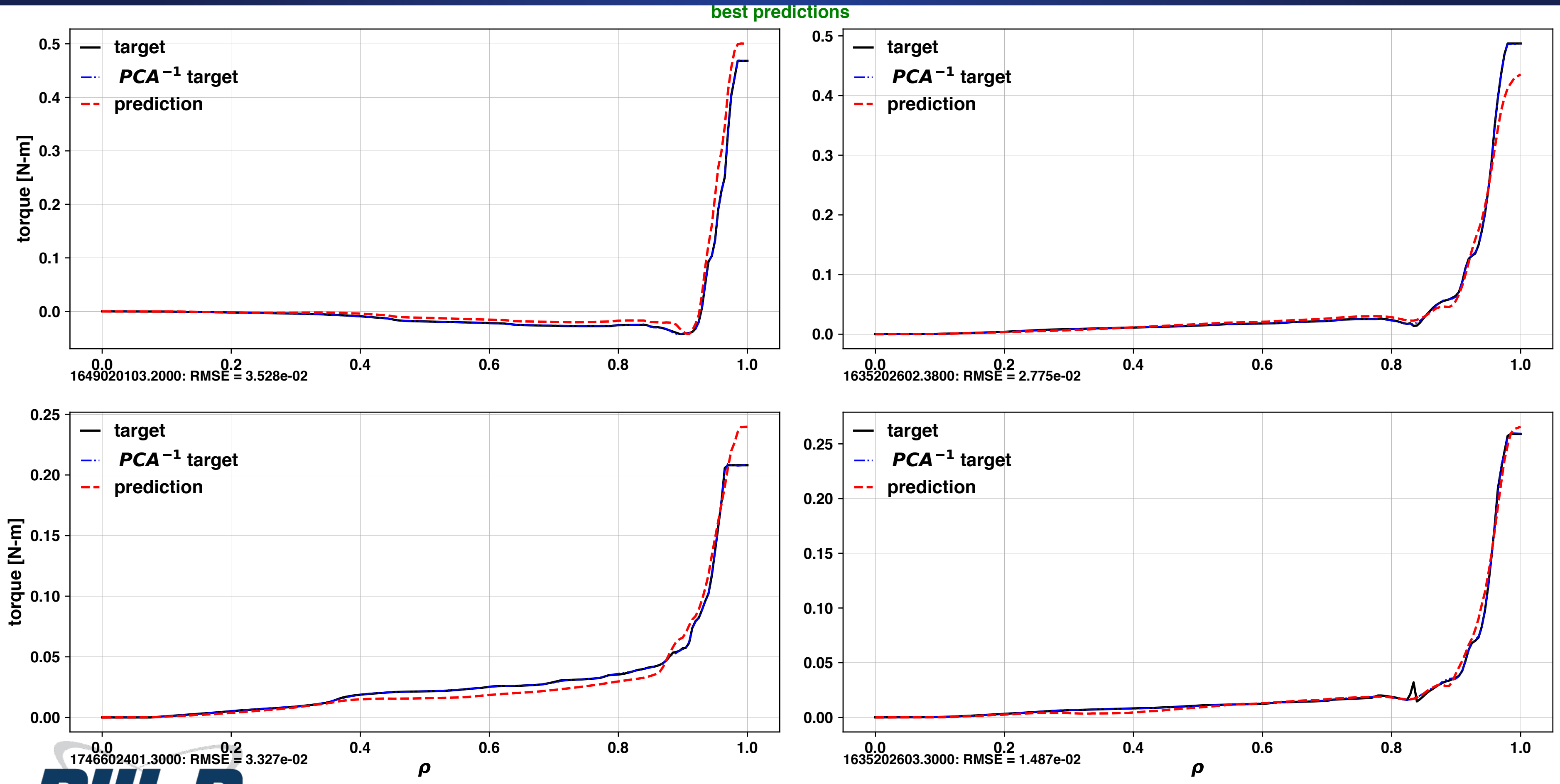


GPECNet T_{coil} profile regression uses RMSE to gauge model performance

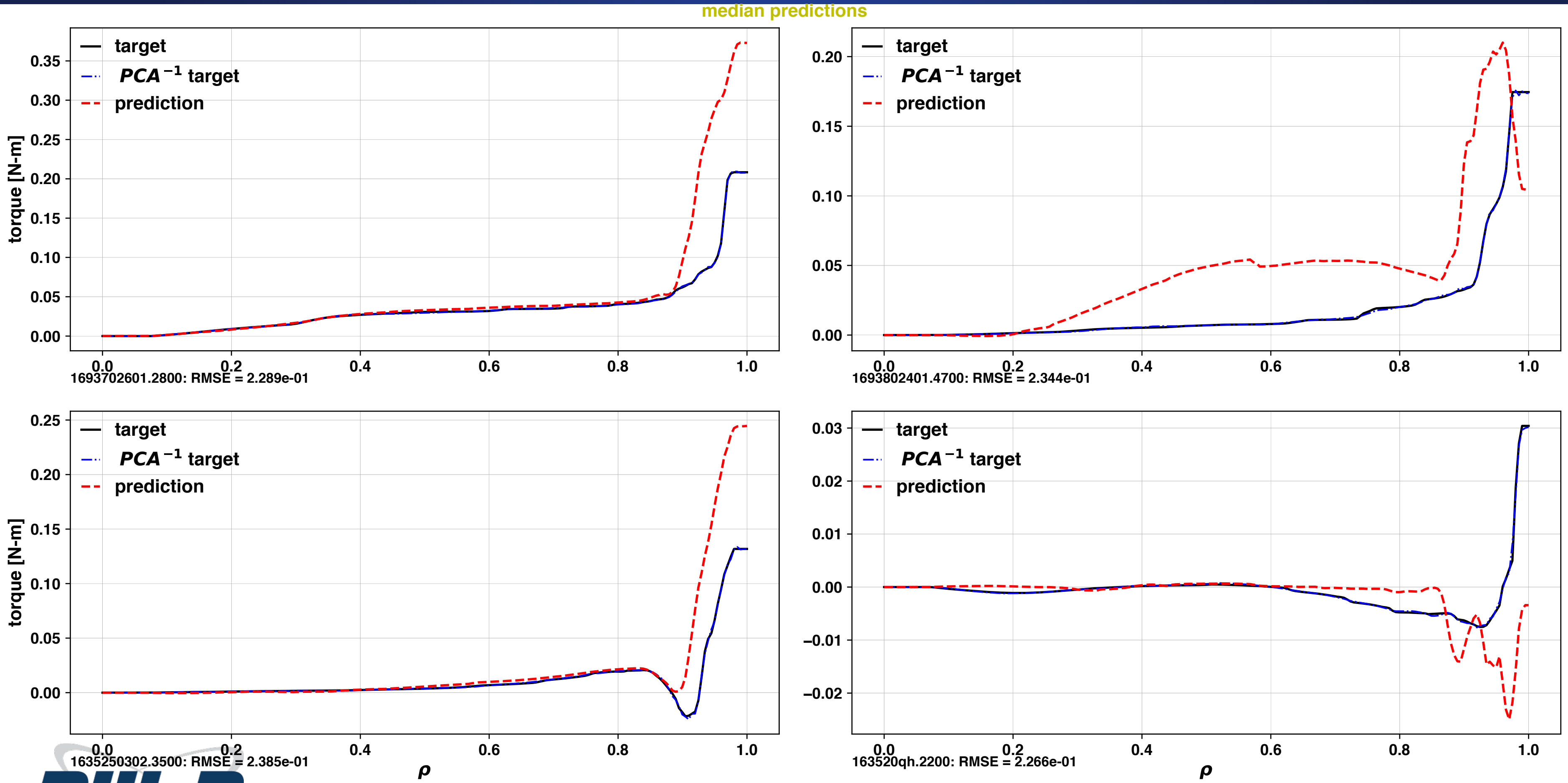
worst predictions



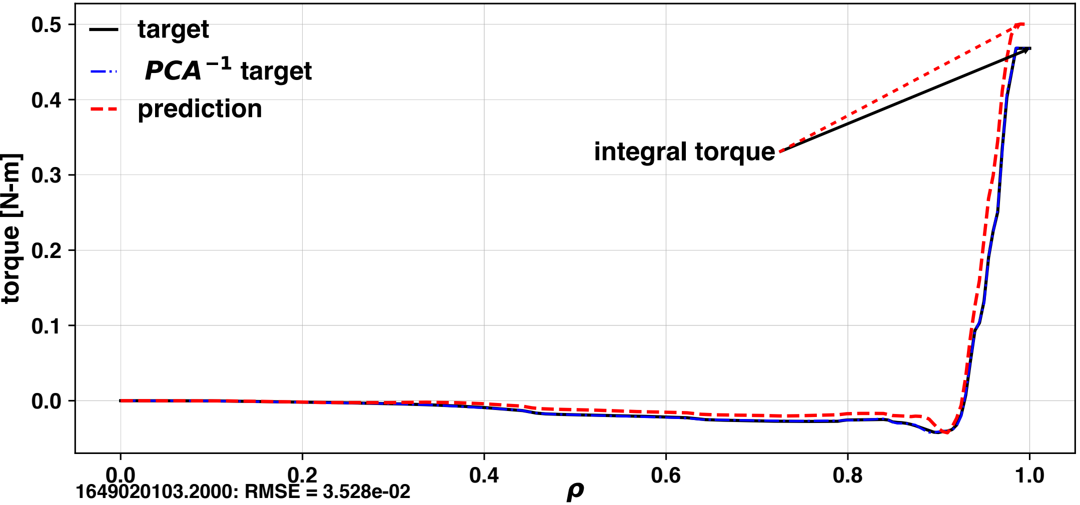
GPECNet T_{coil} profile regression uses RMSE to gauge model performance

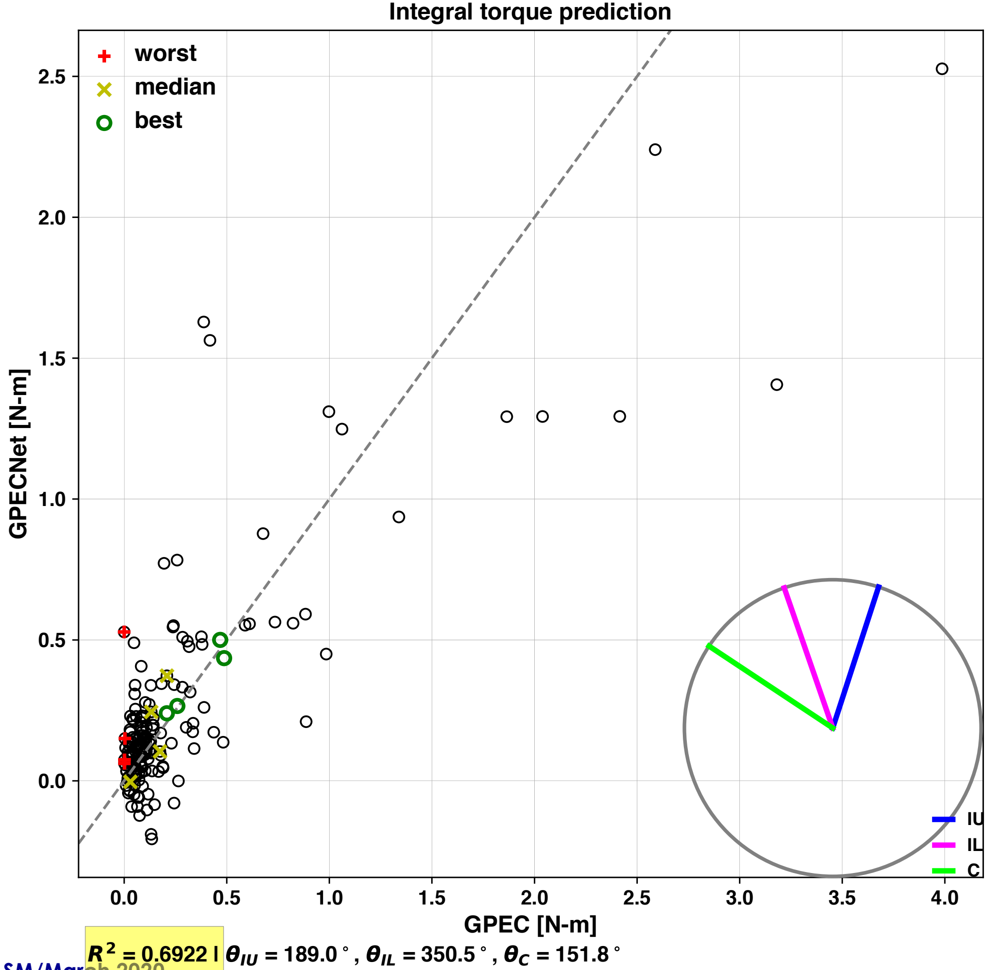


GPECNet T_{coil} profile regression uses RMSE to gauge model performance

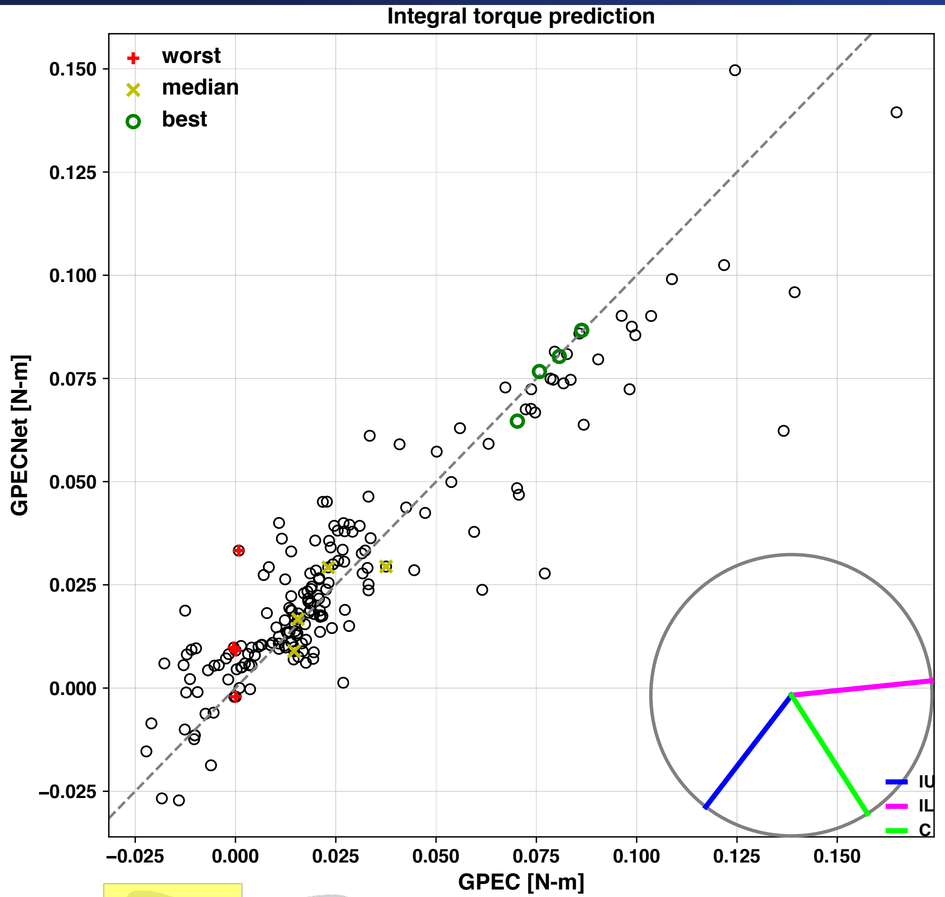


Linear regression plot of edge integral torque provides a scalar result to gauge performance

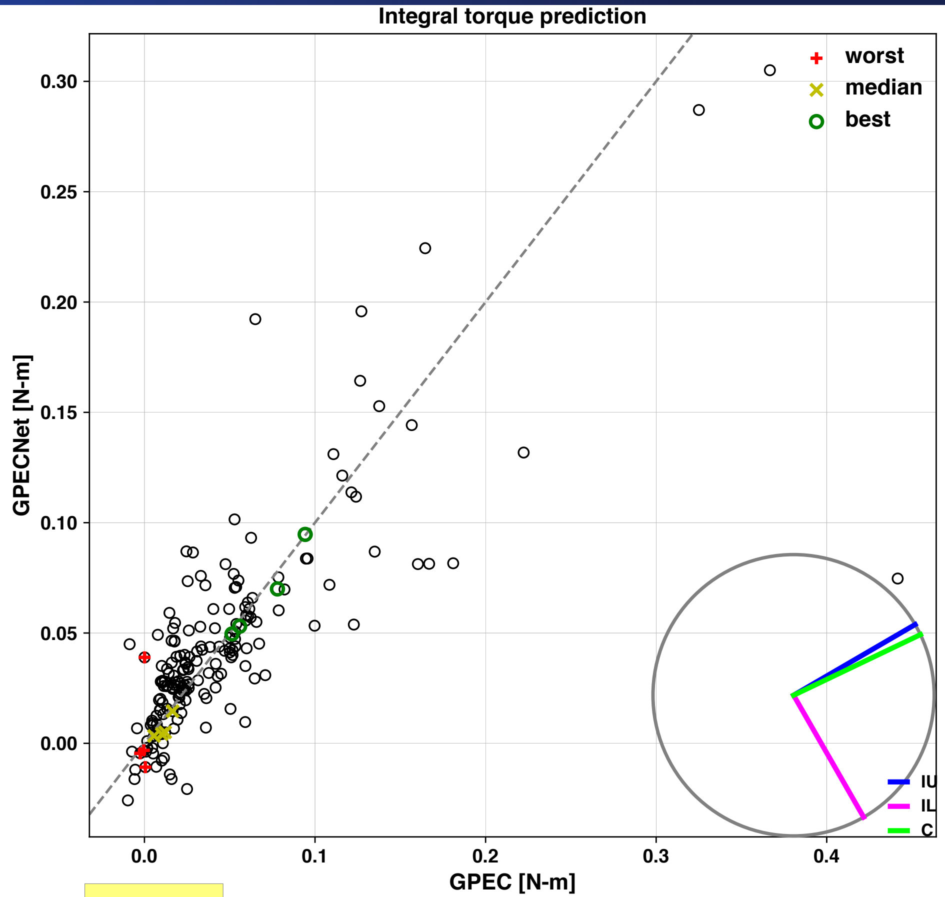




Quality of regression fit depends heavily on coil current relative phase



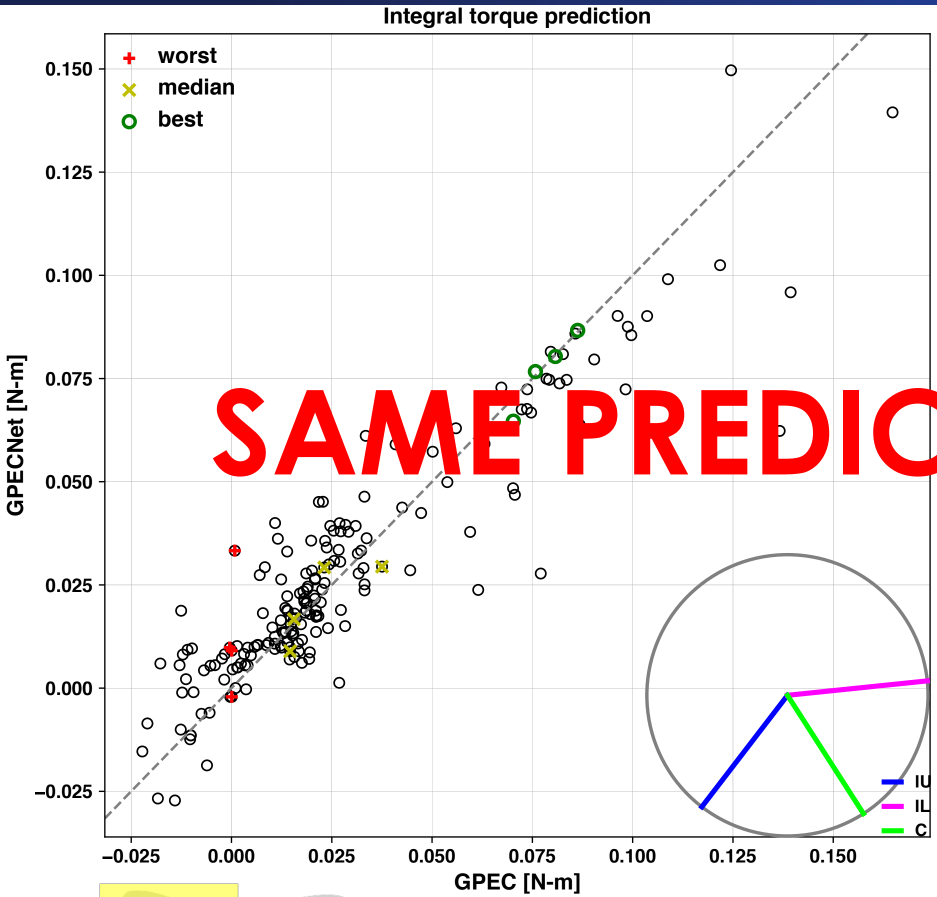
R^2 = 0.8534 | θ_{IU} = 217.5°, θ_{IL} = 84.1°, θ_C = 147.3°

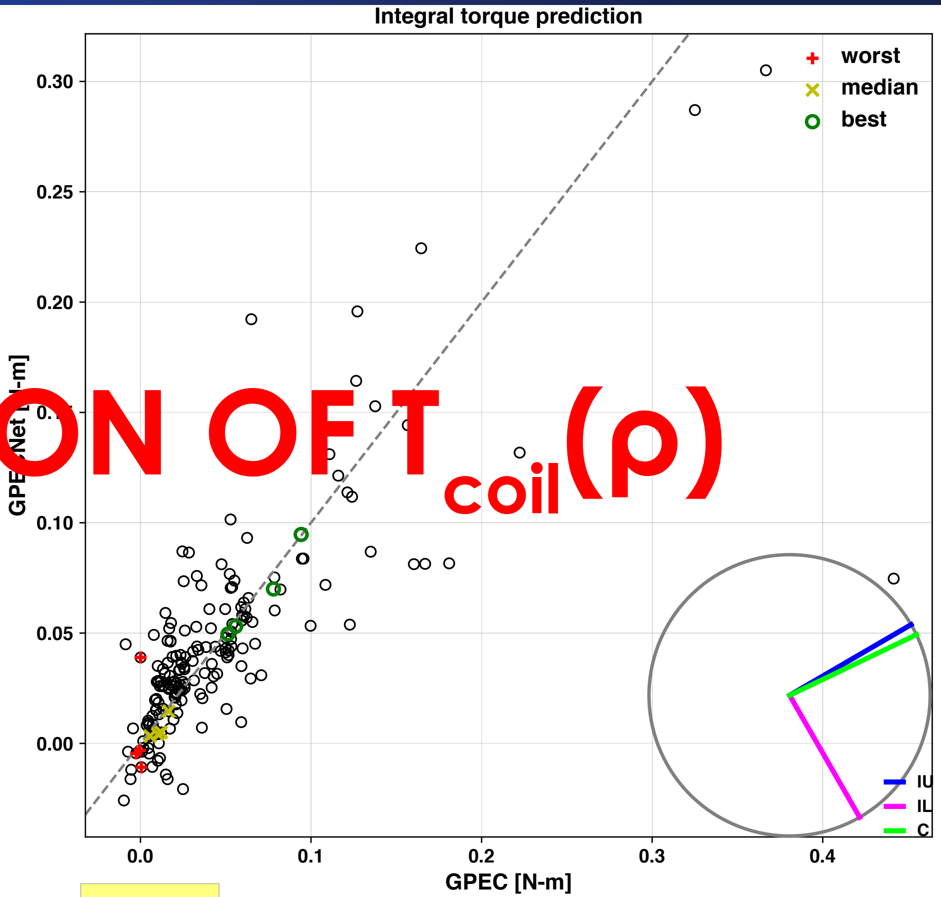


R^2 = 0.5830 | θ_{IU} = 29.9°, θ_{IL} = 75.1°, θ_C = 32.2°

DIII-D NATIONAL FUSION FACILITY

Quality of regression fit depends heavily on coil current relative phase





SAME PREDICTION OF $T_{coil}(\rho)$

$R^2 = 0.8534$ | $\theta_{IU} = 217.5^\circ$, $\theta_{IL} = 84.1^\circ$, $\theta_C = 147.3^\circ$

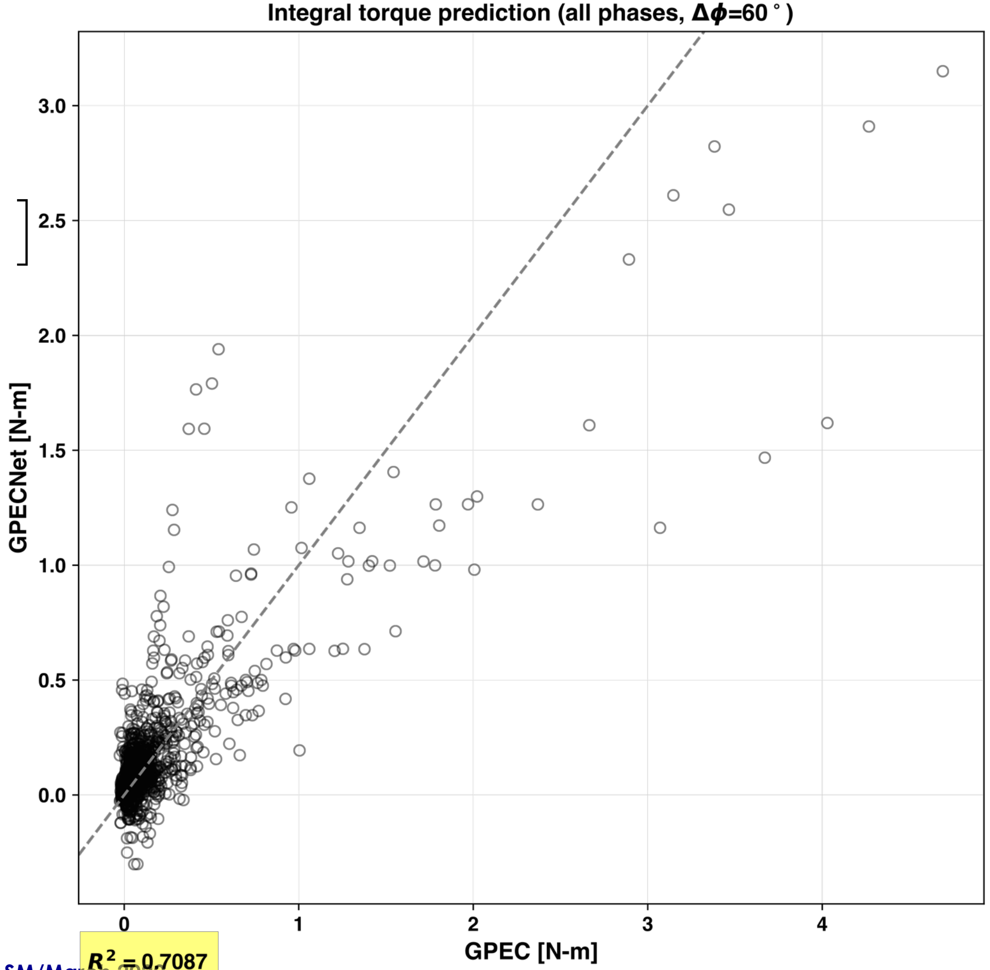
$R^2 = 0.5830$ | $\theta_{IU} = 29.9^\circ$, $\theta_{IL} = 75.1^\circ$, $\theta_C = 32.2^\circ$

Integral torque for all possible phases shows neural net performance

- **Compute all possible coil phases given these phase angles:**

$$\theta = \begin{bmatrix} 0^\circ & 60^\circ & 120^\circ & 180^\circ & 240^\circ & 300^\circ \end{bmatrix}$$

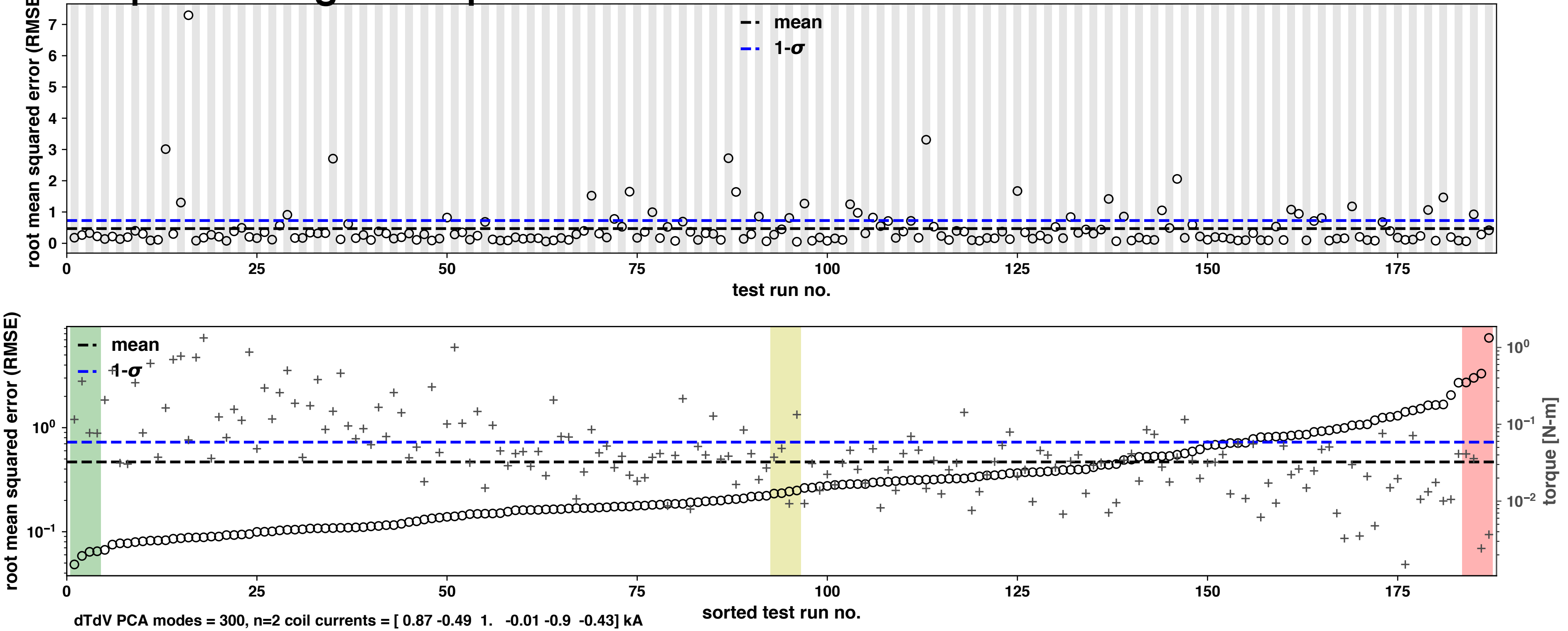
- **C coil phase held constant, IU and IL phase changed relative to C**



R^2 = 0.7087

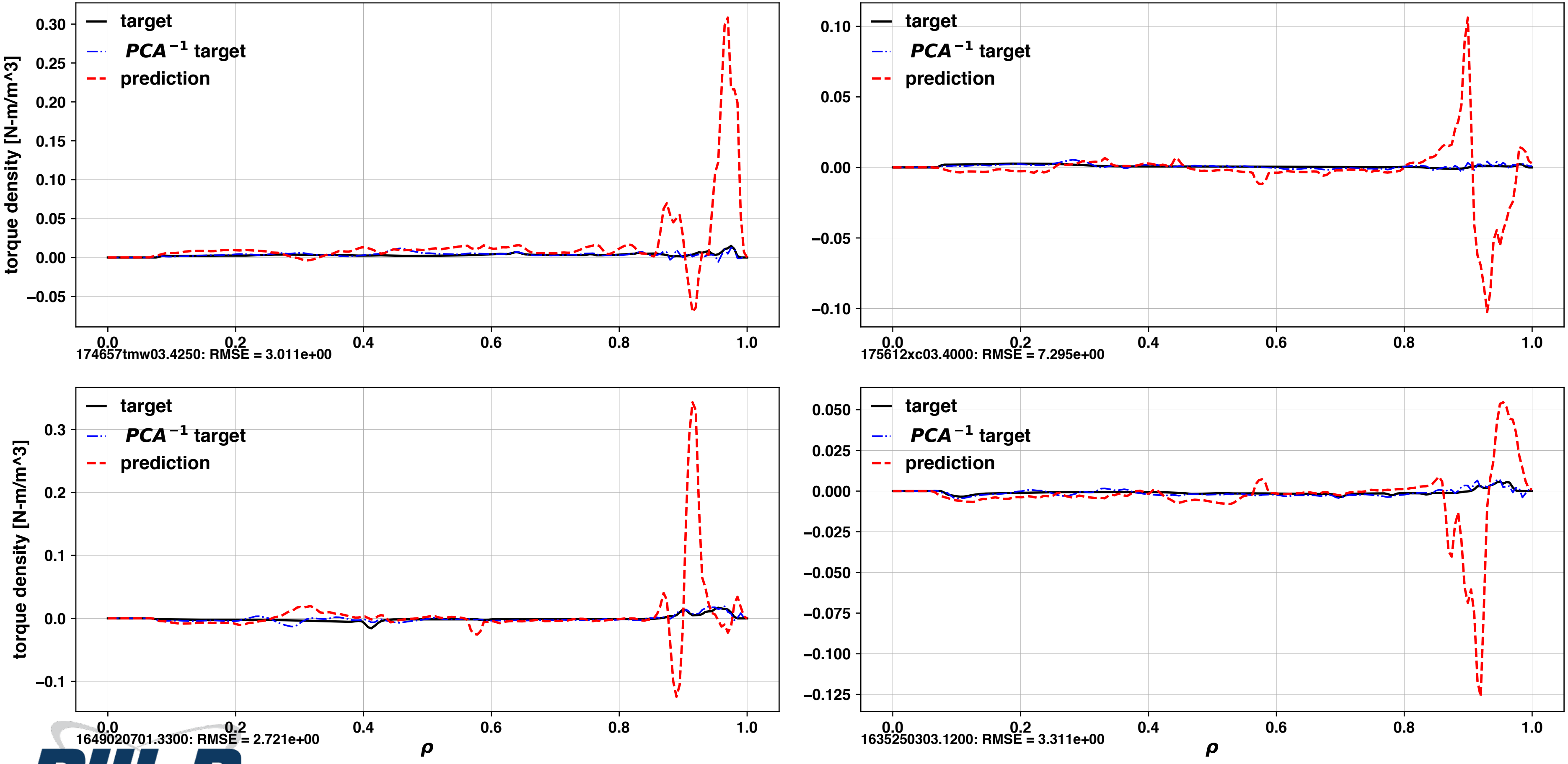
GPECNet dT_{coil}/dV profile also uses RMSE to gauge model performance

- **NN predicts higher torque cases with less RMSE**



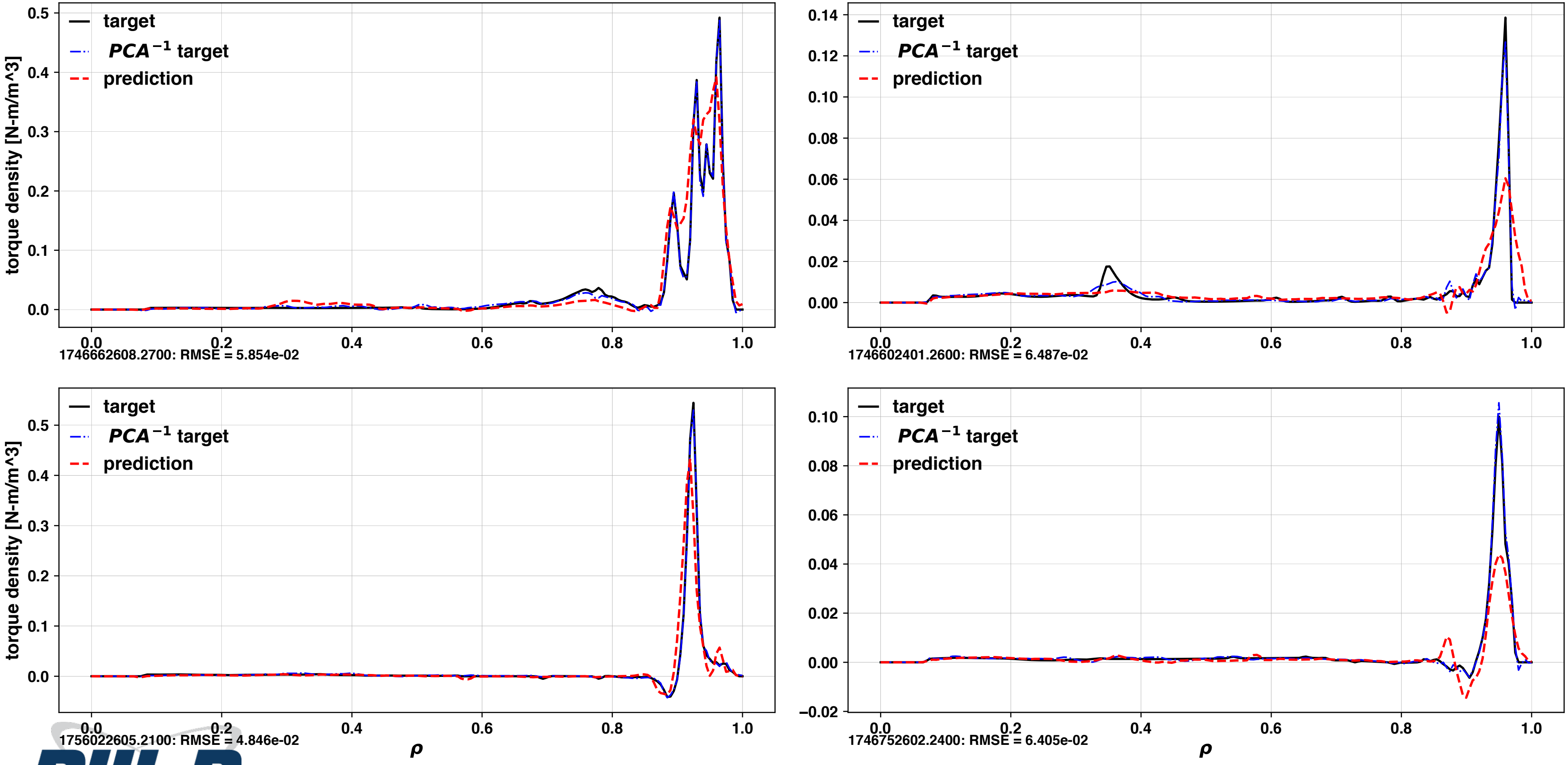
GPECNet dT_{coil}/dV profile also uses RMSE to gauge model performance

worst predictions



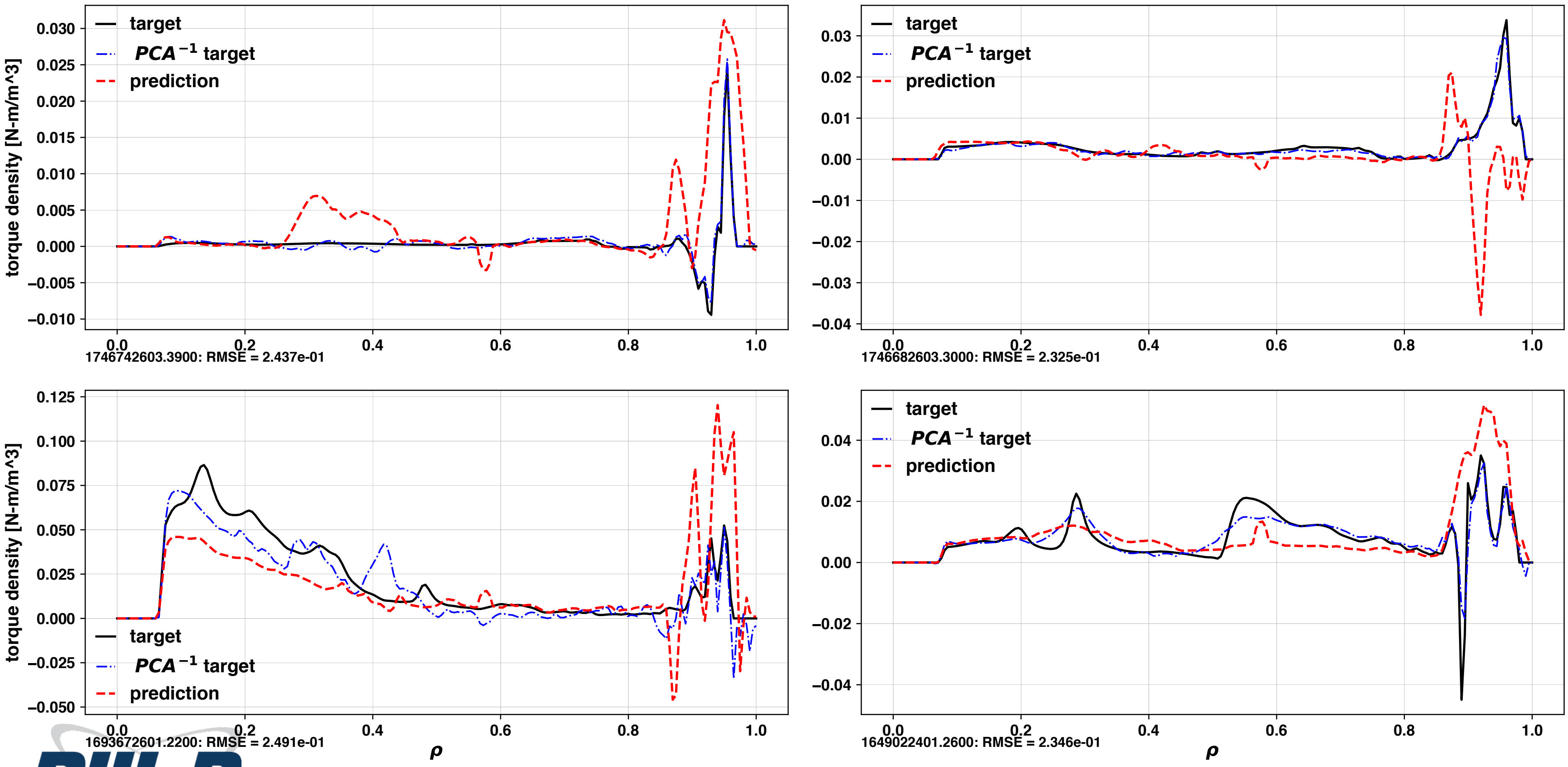
GPECNet dT_{coil}/dV profile also uses RMSE to gauge model performance

best predictions



GPECNet dT_{coil}/dV profile also uses RMSE to gauge model performance

median predictions

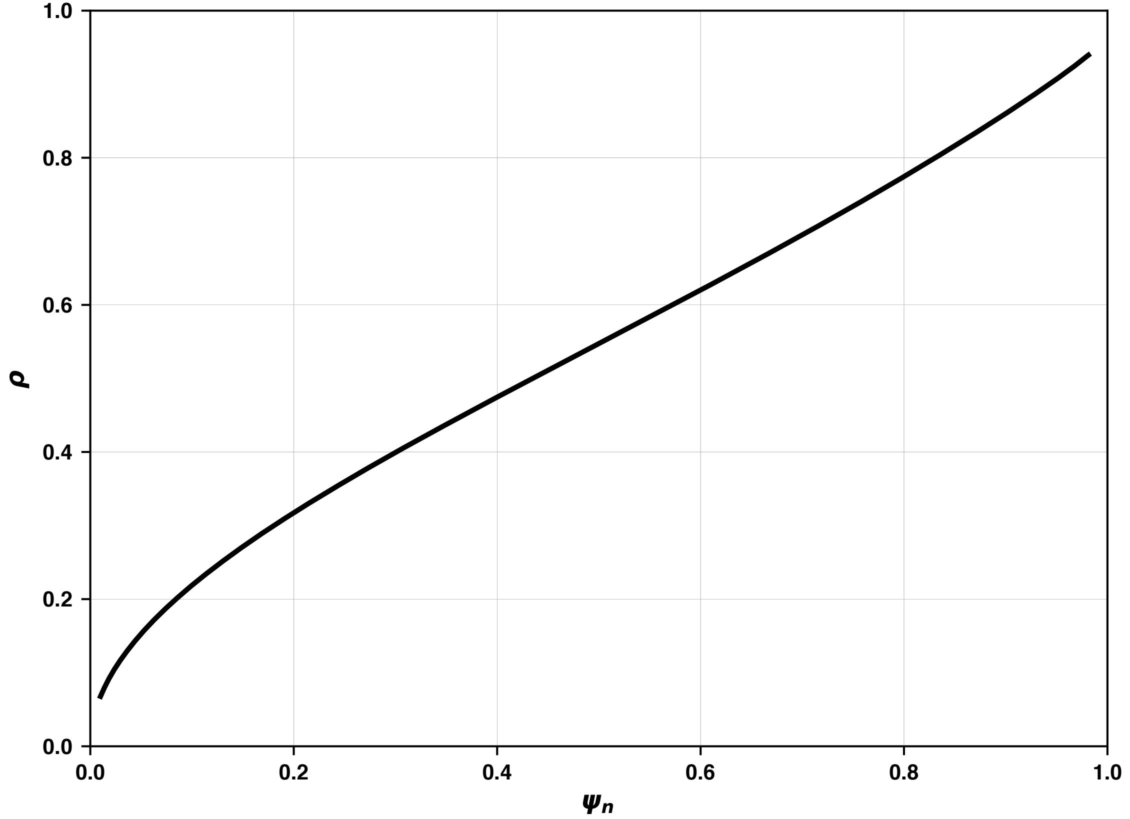


Next steps and future work

- **Neural Net architecture optimization for dT_{coil}/dV**
  - Optimization metric
  - Layer/node size scan
- **dT_{coil}/dV profile output for use in rotation profile controller**
  - Model Predictive Controller (MPC) using torque density estimates from GPECNet and NUBEAMNet neural network models
  - Evaluate controller performance in TRANSP
- **Adapt NUBEAMNet PCS code for use with GPECNet**
  - For use in NTV experiments at DIII-D

General Perturbed Equilibrium Codes (GPEC) calculated torque matrix profile

- GPEC provides mapping from ψ_n to ρ

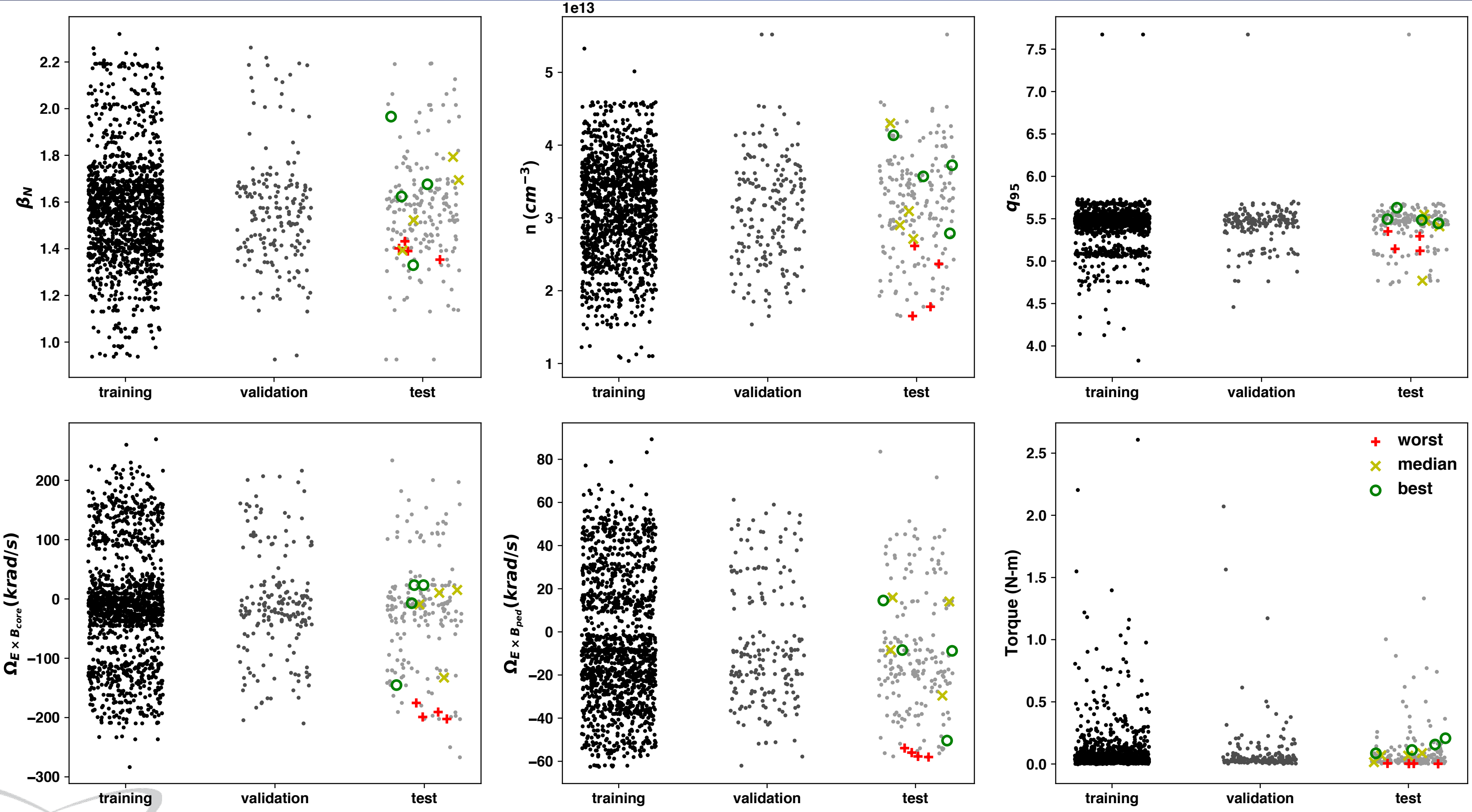


DIII-D NATIONAL FUSION FACILITY

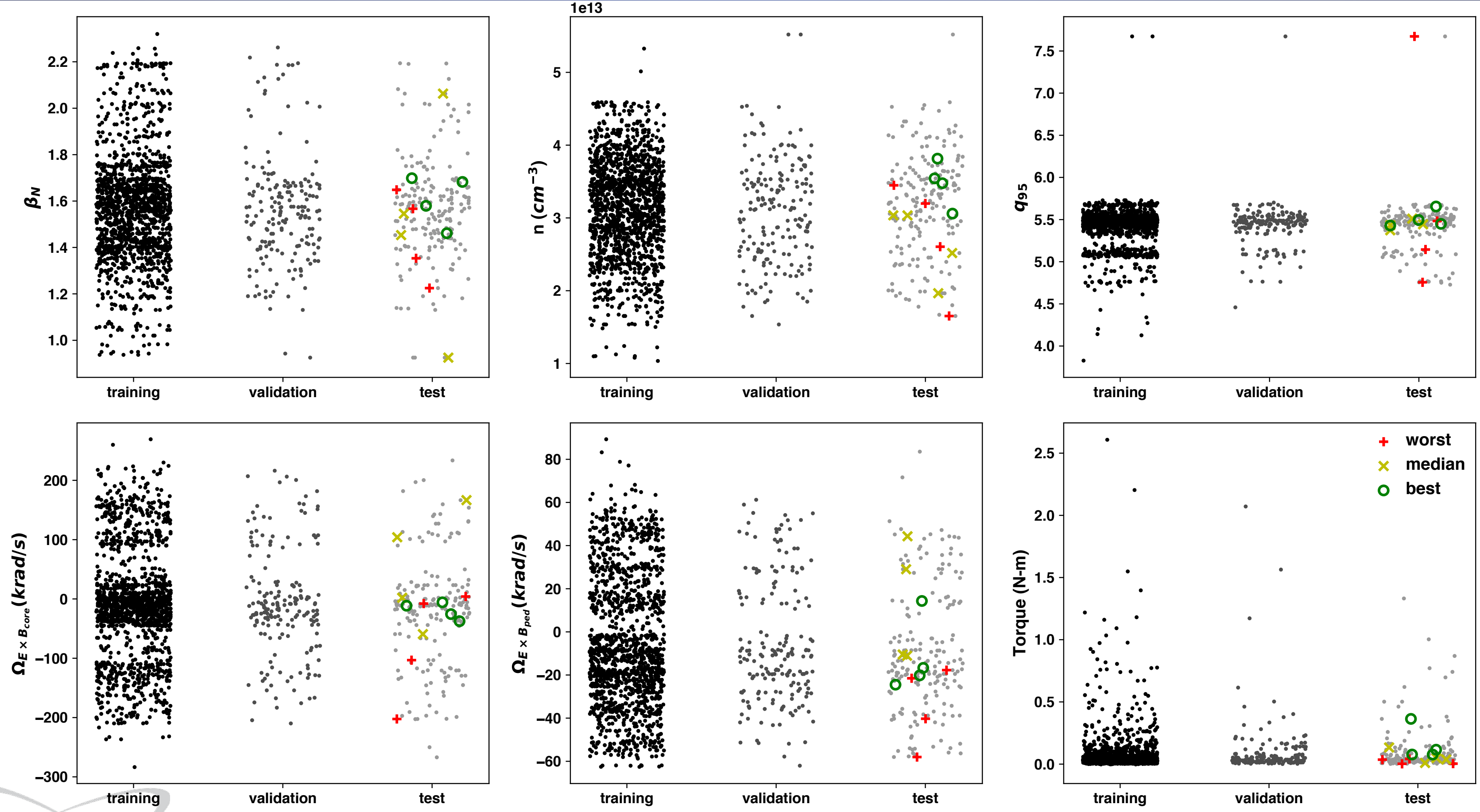
GPEC

http://princetonuniversity.github.io/GPEC/

GPECNet T_{coil} profile regression uses RMSE to gauge model performance



GPECNet dT_{coil}/dV profile also uses RMSE to gauge model performance



DIII-D NATIONAL FUSION FACILITY

T_{coil} Neural Net Keras.model.summary()

```
Model: "sequential_128"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_509 (Dense)            (None, 100)               3000
_________________________________________________________________
dense_510 (Dense)            (None, 200)               20200
_________________________________________________________________
dense_511 (Dense)            (None, 300)               60300
_________________________________________________________________
dense_512 (Dense)            (None, 300)               90300
=================================================================
Total params: 173,800
Trainable params: 173,800
Non-trainable params: 0
_________________________________________________________________
```

δW_0 Neural Net Keras.model.summary()

```
Model: "sequential_114"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_453 (Dense)            (None, 100)               3000
_________________________________________________________________
dense_454 (Dense)            (None, 50)                5050
_________________________________________________________________
dense_455 (Dense)            (None, 33)                1683
_________________________________________________________________
dense_456 (Dense)            (None, 1)                 34
=================================================================
Total params: 9,767
Trainable params: 9,767
Non-trainable params: 0
_________________________________________________________________
```