Offline Model-Based Reinforcement Learning for Tokamak Control

Ian Char[1], **Joseph Abbate**[2], **László Bardóczi**[3], **Mark D. Boyer**[2], **Youngseog Chung**[1], **Rory Conlin**[4], **Keith Erickson**[2], **Viraj Mehta**[5], **Nathan Richner**[6], **Egemen Kolemen**[2,4], and **Jeff Schneider**[1,5]

[1]Machine Learning Department, Carnegie Mellon University
[2]Princeton Plasma Physics Laboratory
[3]General Atomics
[4]Department of Mechanical and Aerospace Engineering, Princeton University
[5]Robotics Institute, Carnegie Mellon University
[6]Oak Ridge Associated Universities

Editors: N. Matni, M. Morari, G. J. Pappas

Abstract

Control for tokamaks, the leading candidate technology for nuclear fusion, is an important pursuit since the realization of nuclear fusion as an energy source would result in virtually unlimited clean energy. However, control of these devices remains a challenging problem due to complex, non-linear dynamics. At the same time, there is promise in learning controllers for difficult problems thanks to recent algorithmic developments in reinforcement learning. Because every run (or *shot*) of the tokamak is extremely expensive, in this work, we investigated learning a controller from logged data before testing it on a tokamak. In particular, we used 18 years of data from the DIII-D device in order to learn a controller for the neutral beams that targets specified β_N (normalized ratio of plasma pressure to magnetic pressure) and rotation quantities. This was done by using the data to first learn a dynamics model, and then by using this model as a simulator to generate experience to train a controller via reinforcement learning. During a control session on DIII-D, we tested both the ability for our dynamics model to design feedforward trajectories and the controller's ability to do feedback control to achieve specified targets. This work marks some of the first steps in doing reinforcement learning for tokamak control through historical data alone.

Keywords: Reinforcement Learning, Model-Based Reinforcement Learning, Tokamak Control

1. Introduction

Unlocking the potential of nuclear fusion as an energy source would have profound impacts on the world. Nuclear fusion is an attractive energy source since the fuel is abundant, there is no risk of meltdown, and there are no high-level radioactive byproducts (Walker et al., 2020). Perhaps the most promising technology for harnessing nuclear fusion as a power source is the tokamak: a device that relies on magnetic fields to confine a toroidal plasma. While strides are being made to prove that net energy output is possible with tokamaks (Meade, 2009), there are still crucial control challenges that exist with these devices (Humphreys et al., 2015).

At the same time, exciting developments in reinforcement learning (RL) have provided the possibilities for learning complex controls. While there have been some astounding results that leverage RL, they depend either on a cheap, accurate simulator or an expensive set up where many

samples can be collected on the actual device. In our setting, unfortunately, it is infeasible to collect enough samples on the real device, and simulators are both expensive and do not reflect the true dynamics for many aspects of the plasma. Thus, in this work we focused learning controls entirely from historical data. In particular we learned controls for DIII-D, a device operated by General Atomics in San Diego, California. This device has been in operation since 1986, during which there have been over one hundred thousand "shots" (runs of the device). We use approximately 15k of these shots to learn dynamics models that predict the evolution of the plasma, subject to different actuator settings. These surrogate models can then be used as a simulator that generates experience for the RL algorithm to train with. We applied this method to train a controller that uses DIII-D's eight neutral beams to achieve desired β_N (the normalized ratio between plasma pressure and magnetic pressure) and differential rotation targets.

In the following, we first give an overview of the state and actuator variables considered for this control task and the training procedure for learning the controller. We then review our validation experiments conducted on DIII-D for both feedforward and feedback control. The results show the effectiveness of using the learned dynamics models for feedforward control, and while we found feedback control to be more challenging, our controller showed clear promise for β_N tracking. We believe this is the first work for doing offline RL for feedback control on a tokamak, and one of the first works to apply offline reinforcement learning to an expensive device. Thus, we end this paper with a discussion of perspectives on the offline RL problem gleaned from this application.

2. Related Work

Reinforcement Learning. Several advancements in the field of deep RL have made the prospect of doing continuous control within reach. Strides in both on-policy algorithms (Schulman et al., 2015, 2017; Mnih et al., 2016) and off-policy algorithms (Lillicrap et al., 2015; Fujimoto et al., 2018; Haarnoja et al., 2018) have resulted in relatively stable optimization procedures that can produce controls for complex, high-dimensional problems. However, these "model-free" methods are data hungry and usually require millions of samples from the environment. To address this, "model-based" reinforcement learning (MBRL) algorithms can often learn to control with fewer samples by simultaneously learning a model of the dynamics. These models can either be used for better estimates of the value function (Feinberg et al., 2018; Amos et al., 2021) or can be used to generate additional, fictitious data for the agent to train on (Kurutach et al., 2018; Janner et al., 2019). We use the latter MBRL approach in this work.

The aforementioned developments in RL target the standard *online* setting, in which agents gather experience through interactions with the environment. In contrast, *offline* RL (Levine et al., 2020) attempts to learn a policy only through logged, historical interactions from possibly many different policies. This is an attractive setting since many real-world problems will have logged interactions to leverage; however, the added restriction usually causes deep RL algorithms designed for the online setting to fail because they pick actions that are out of distribution. To combat this problem, offline RL algorithms add in extra penalization to ensure that the optimization procedure chooses actions close to the support of the dataset (Kumar et al., 2020; Wu et al., 2019; An et al., 2021). There have also been a number of offline MBRL algorithms which rely on the uncertainty in the dynamics models for penalization (Yu et al., 2020; Kidambi et al., 2020; Yu et al., 2021). In our work we decided against using a penalization scheme for a couple of reasons. First, the amount of penalization needs to be tuned by evaluating the controller on a real device, something that we do not have the luxury of. Second, the dynamics models in these works are only accurate for a few time

steps, a problem that usually plagues autoregressive models due to the multiplicative accumulation of error (Asadi et al., 2018). For example, in Yu et al. (2020) the model is only used for 1 or 5 time steps (see Appendix G). We believe our setting is unique since we are able to learn a dynamics model that is often accurate for entire shots.

Learning Controls for Tokamaks There has recently been a surge of interest in applying machine learning for controls of tokamaks. Many of these works focus on predicting disruptions for avoidance or safe shutdowns (Fu et al., 2020b; Parsons, 2017; Rea et al., 2019; Boyer et al., 2021); however, in this work, we focus on control during stable operation. Under these conditions, Char et al. (2019) used contextual Bayesian optimization to find controls that balance increasing β_N and keeping the plasma stable. While this technique is fully automated, Baltz et al. (2017) present an algorithm that performed human-in-the-loop optimization to increase plasma confinement.

In terms of modeling dynamics, Abbate et al. (2021) used a convolutional neural network to model the evolution of the plasma's profiles, and they later used this model for control via MPC (Abbate et al., 2023). Many of the choices for our dynamics modeling, such as the signals to use and the data preprocessing, were directly inspired by this work. Additionally, Seo et al. (2021, 2022) learned a dynamics model for the KSTAR tokamak. They then used RL for tracking several scalar values including β_N; however, they used this policy to generate feedforward controls only. While Wakatsuki et al. (2021) trained a feedback controller to do ion temperature gradient control for the JT-60 tokmak, this controller was both trained and tested in the same TOPICS simulator. To the best of our knowledge, the only RL feedback controller deployed on a real device up to this point was done by Degrave et al. (2022). They leveraged a simulator to learn a controller for the plasma's shape on Tokamak a Configuration Variable (Coda et al., 2019). Our work differs not only in the goal and actuators used, but also from the fact that we leveraged historical data exclusively. The dynamics for the plasma's shape is more well-understood than other aspects of the plasma, and as such, can be modeled and controlled relatively precisely (Walker et al., 2020, 1997). While the potential impact of learning controls for other aspects of the plasma is great, the corresponding simulations are expensive and less precise, which prompted us to leverage logged data.

3. Method

Problem Description We cast the problem of control of the tokamak as a discrete-time, infinite-horizon Markov decision process (MDP). In particular, let $\mathcal{M} := \langle \mathcal{S}, \mathcal{A}, \gamma, T, r, \rho \rangle$ be the MDP, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\gamma \in (0, 1)$ is the discount factor, $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and ρ is the initial state distribution. Lastly, $T : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$ is the transition function, where $\mathcal{P}(\mathcal{S})$ denotes the space of distributions over $\mathcal{S}$. Each transition corresponds to a 100ms time step in real time. In practice, it is difficult to observe all state variables in real-time for feedback control. As such, we learn a policy for the partially observable MDP (POMDP). Let $\mathcal{O}$ be the observation space and let $h : \mathcal{S} \to \mathcal{O}$ be the mapping from states to observations. The overall goal is then to learn a policy $\pi : \mathcal{O} \to \mathcal{P}(\mathcal{A})$ that maximizes the expected discounted sum of rewards $\mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t)\right]$, where $s_0 \sim \rho$, $a_t \sim \pi(h(s_t))$, and $s_{t+1} \sim T(s_t, a_t)$.

State and Action Spaces DIII-D has a number of actuators for controlling the plasma. Of key interest to our work are the eight neutral beams that inject particles into the core of the plasma (see Figure 1) (Grierson et al., 2021). These are used to inject both power and torque into the plasma, and

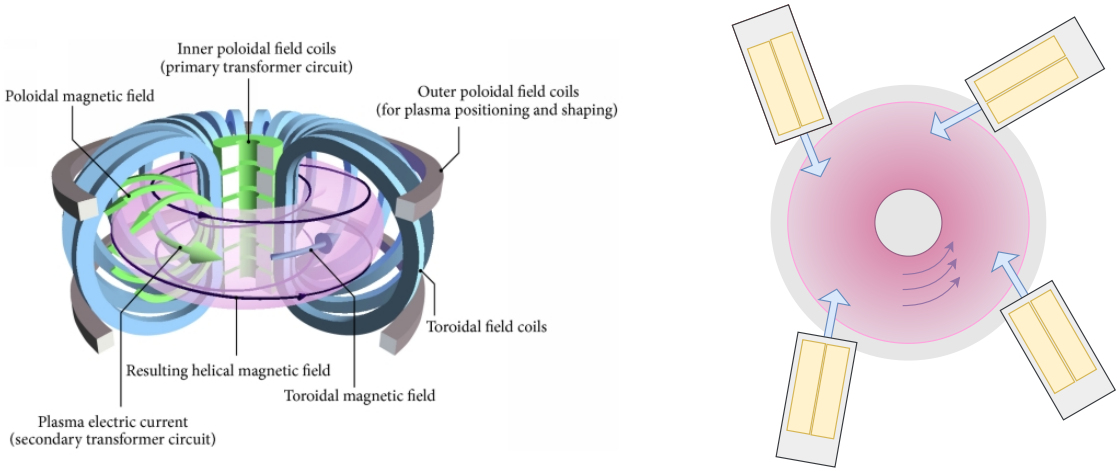

Figure 1: **Diagram of Tokamak (Left) and Top-Down View of DIII-D (Right)**. Looking at the right figure, one can see that each beam line contains two independent neutral beam sources (yellow boxes). Here, the plasma is rotating in the counter-clockwise direction, and the two beams in the bottom left of the figure are oriented to be counter-current. Because of this, the total power and torque are decouple. The left image is from Li et al. (2014). The right image gives a rough idea of the beam positioning and is not drawn to proportion.

because two of these beams can be oriented in the opposite toroidal direction, the amount of total power and torque injected can be decoupled. We also consider the the ohmic coil (for controlling current), gas valves (for controlling plasma density), and toroidal field coils for our modelling (Luxon, 2002). Lastly, one can also control the shape of the plasma via the field coils. We assume that these controls are sophisticated enough to the point that we can control the elongation, top triangularity, bottom triangularity, and the minor radius of the plasma exactly. While there are many other ways of affecting the plasma, this subset encapsulates most standard runs on the device. All of these actuators could potentially be part of the action space; however, we focused on only the neutral beams for this work. As such, the action space is simply the total power and torque injected from the neutral beams.

For the state space, we assume that the plasma can be fully characterized by the current settings of the above actuators, three scalar values, and five "profiles" which consist of discretized measurements of physical quantities along the minor radius of the toroid. The scalar states consist of the line-averaged electron density, the internal inductance, and β_N, which is the normalized ratio between plasma pressure and magnetic pressure. β_N is an important quantity as it can be used as a rough economic indicator of efficiency. Radial profiles of the ion rotation, pressure, electron temperature, electron density, and the safety factor, known as "q", are also used. The q profile is the number of toroidal transits per poloidal transit of a magnetic fieldline, and is an important indicator of stability of the plasma (the higher the q factor the better). Following Boyer and Chadwick (2021), we reduced the dimensionality of these profile states (originally 64 dimensions) by using principal component analysis (PCA). We found that we can explain 99% of the variance in the data by using two principal components for the q and pressure profiles, and by using four components for the rest of the profiles. For these scalar and profile descriptions of the plasma, we assume the state can be represented by the current measurements as well as the measurements 100ms in the past. This assumption stems decisions made about the learned dynamics model. In particular, including this history increased the predictive performance of the dynamics model; however, including history for the actuators variables made the model susceptible to overfitting).

			MDP Spaces		
Signal Group	**Signals**	**Actuator**	**State ($\mathcal{S}$)**	**Action ($\mathcal{A}$)**	**Observation ($\mathcal{O}$)**
Scalar States	β_N	✗	✓	✗	✓
	li (Internal Inductance)	✗	✓	✗	✗
	Line Averaged Density	✗	✓	✗	✗
Profile States	Ion Rotation, Electron Density, Electron Temperature, Pressure, q	✗	✓	✗	✗
Neutral Beam Variables	Power Injected and Torque Injected	✓	✓	✓	✓
Shape Variables	Elongation, Top Triangularity, Bottom Triangularity, a_{minor}	✓	✓	✗	✗
Other Actuators	Current Target, Density Target, and Toroidal Field	✓	✓	✗	✗
Observations	DR, DR Target, and β_N Target	✗	✗	✗	✓
Total Dimensions		9D	47D	2D	10D

Table 1: **Overview of Signals**. Note that DR and both targets are not in the state space. DR is calculated from rotation and q (but is not modeled explicitly), and the targets do not influence the transition function. The total dimension of the state space factors in the number of PCA components used to represent the profiles, and the total dimensions of both state and observation spaces accounts for measurements 100ms in the past. While the state dimension is relatively high, the dynamics model only predicts future scalar and profile state variables (19D).

Objective and Reward Function The control objective is to do target tracking for two quantities: β_N and *differential rotation* (DR). Specifically in this work, DR refers to the difference in the rotation profile at the locations where $q = 1$ and $q = 2$ (see Figure 2). This is an important quantity of interest as it is hypothesized that higher DR results in a more stable plasma (Bardoczi et al., 2021; Tobias et al., 2016; Buttery et al., 2008; Reimerdes et al., 2007; Politzer et al., 2008). While control for β_N is relatively straight forward, DR relies on the correct measurements of two profiles and is therefore harder to predict and control. For every episode in the MDP, new targets β'_N and DR' are drawn from target distributions. In particular, $\beta'_N \sim \mathcal{U}(1.25, 2.5)$ and $DR' \sim \mathcal{U}(10, 80)$. The reward at time step t, $r(s_t)$, is then

$$\frac{-1}{C_1}\left(\beta_N^{(t)} - \beta'_N\right)^2 + \frac{-1}{C_2}\left(DR^{(t)} - DR'\right)^2,$$

where $\beta_N^{(t)}$ and $DR^{(t)}$ are the current measurements at time t, and C_1 and C_2 are positive normalizing constants that puts each term onto the same scale.

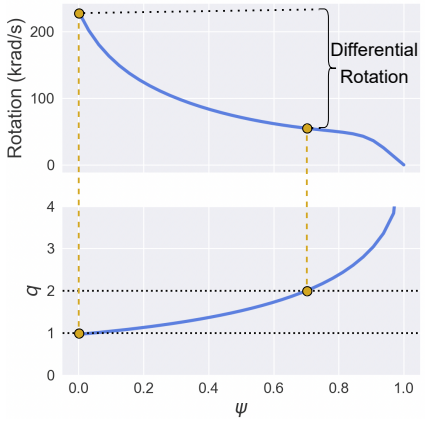

Figure 2: **Visual Representation of Differential Rotation (DR)**. The top and bottom plots show examples of rotation and q profiles, respectively. The q profile dictates the ψ (flux surface) locations to measure on the rotation profile.

3.1. The Dynamics Model and Controller

Dynamics Model We chose to approximate $\mathcal{T}$ using a fully connected neural network which takes in the current state of the system and the actuators settings planned for 100ms in the future, and then outputs predictions for the (non-actuator) state variables 100ms into the future. We trained this network on a dataset consisting of 15,534 shots (or 268,702 time steps), which was pre-processed

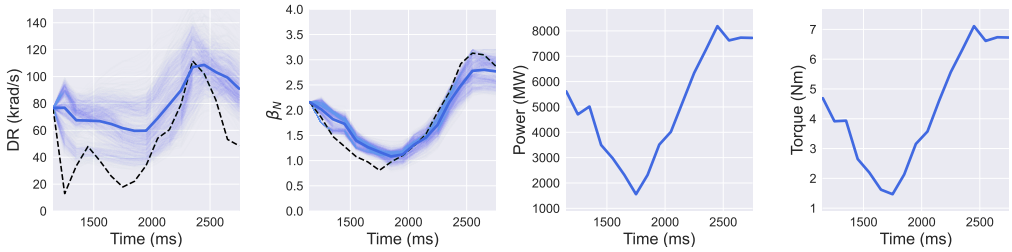

Figure 3: **Replay of Shot 187076.** Here, the model receives the first observations of β_N and DR, but then autoregressively predicts these values into the future. The faded lines are different samples of the neural network parameterization, and the solid lines are the average over the different predictions. Note that there are no faded lines for the power and torque plots since actuators are given to the model and are not predicted. The black dashed lines are the real observations.

in the same manner as Abbate et al. (2021). In particular, at each 100ms increment we formed the observation for each signal by averaging over every measurement 25ms previous to that point.

To ensure that a controller learned on this model will still perform on the real device, it is important to learn many different possibilities of what the dynamics could be. This has been shown to be essential both in the context of MBRL (Chua et al., 2018), and in the context of doing "sim2real" (e.g. domain randomization (Tobin et al., 2017)). We incorporated uncertainty by learning a subspace of network parameters (Wortsman et al., 2021; Benton et al., 2021), which has been shown to better calibrated models over standard ensembling. In particular, we used an ensemble of five networks to learn a simplex of network parameters following the procedure described in Wortsman et al. (2021). We repeated this training procedure five times to learn five different simplices. By making a uniform draw from this collection of network parameters, we can sample a new possibility for the dynamics.

To do hyperparameter tuning and evaluation, we took the most recent 10% of shots as our test set. Our tuning procedure targets high explained variance (EV) for one-step predictions. After performing grid search, we settled on a model with 4 hidden layers of 512 units, and a learning rate of 3e-4. When learning the simplex, we encouraged diversity by adding a cosine similarity penalizer to the loss function (see Wortsman et al. (2021)), and we found that a coefficient of 5 to this penalty gets the best results. Averaged across five seeds and one hundred samples from the simplex for each seed, these mean predictions achieve an EV score of 0.46 for β_N, 0.43 for the first rotation PCA component, and 0.33 averaged across all output signals. We use the Uncertainty Toolbox library Chung et al. (2021) to evaluate our ensemble's predictive uncertainty. We find that our model tends to be overconfident and achieves a miscalibration area of 0.26 for β_N, 0.25 for the first rotation PCA component, and 0.297 averaged across all predictions. We believe that part of the reason these scores are poor is that the future shots in the test set are meaningfully different. However, qualitatively the model often captures the trends of the state quite well. Figure 3 shows the replay for shot 187076, which was not seen during training. This shot is significantly unique from other shots in the dataset in that there is a drastic drop then increase of both power and torque. Lastly, although we do find that our predictions into the future are generally stable, there are rare cases where the prediction error explodes. To mitigate against this, we bound the state of the plasma and the amount that it can change to be between the 2.5% and 97.5% quantiles of the dataset.

Controller This learned dynamics model can be used to generate data for learning a controller. For the start state distribution, ρ, we used a uniform distribution over over the first 500ms of flat top (i.e.

where current stops ramping up and becomes stable) for all shots in the dataset. Since the controller is only allowed to counterfactually set the total power and torque of the neutral beams, all other actions are the same as what happened in the historical shot corresponding to the start state. We trained a controller on these generated shots using the Proximal Policy Optimization (PPO) algorithm (Schulman et al., 2017). The controller is able to observe the targets as well as the current and past values of β_N, DR, power injected, and torque injected. We use a policy and value network with 2 hidden layers consisting of 500 units each, a gradient clipping parameter of $\epsilon = 0.25$, and a learning rate of $3e-4$ for both the policy and value networks. We decided on these hyperparameters by using an off-policy evaluation procedure in which we have two sets of dynamics models: one used for training and one used for evaluation. The only difference between the two sets is that the model used for evaluation was trained using both the training and testing set. Additionally, for the start state selection and actuator replays, we reserved some historical shots for the evaluation period. The final set of policies were trained on the test set of models (with some historical shots still held out), and the model that was ultimately selected for deployment was the one with the best returns on the held out shots.

4. Experiment

To test the controller on the device, we implemented our trained policy in DIII-D's plasma control system (PCS) (Margo et al., 2020). We used the Keras2C library (Conlin et al., 2021) in order to transfer our policy network (originally implemented in PyTorch (Paszke et al., 2019)) to a deterministic subset of C, meaning no dynamic memory allocation, system calls, or use of external libraries. For beam control, we used the algorithm presented in Boyer et al. (2019) to decide the duty cycles of each of the eight beams to hit the requested power and torque targets. For inputs to the policy, we relied on the profile fitting algorithm (Shousha et al., 2023) and the charge exchange recombination (CER) diagnostic system (Gohil et al., 1991). The policy sends requests for updates to the beams roughly every 10 ms. During our testing session, we were able to test the β_N and DR tracking separately. We used shot 164987 (shots are labeled) as a reference shot, and the actuators besides the beams were mostly used from this shot.

Feedforward Control To disentangle the predictive power of the dynamics model from the policy learning procedure, we used the dynamics model only to prepare feedforward control for the neutral beams. In particular, we used the model to evaluate how closely target values are achieved given fixed controls where the power and torque are ramped up to constant values. A two-dimensional grid search was performed to find the constant values corresponding to the highest cumulative rewards averaged over the sampled shots. We used an even larger ensemble of models for this procedure where the additional models have slightly altered inputs and outputs. In particular, we included five additional network simplexes that consider the actuators from the previous time steps as input and five network simplexes that only consider quantities relevant for this task as inputs and outputs, i.e. β_N, rotation, q, and beam information. This results in a total of 15 different network simplexes, each composed of 5 networks. We found the performance of each type of model is dependent on the shot, so we used all models in the hope for the most robust solution.

We pick a target $\beta_N = 1.75$ and try to push differential rotation to be high by setting the target to $DR = 40$ krad/s, which is relatively high for this reference shot. Our optimization procedure found that setting the total power to 3.6 MW and the total torque to 2.1 Nm was best. As shown in Figure 4, the choice of these actuators resulted in hitting the β_N target remarkably well. Although the

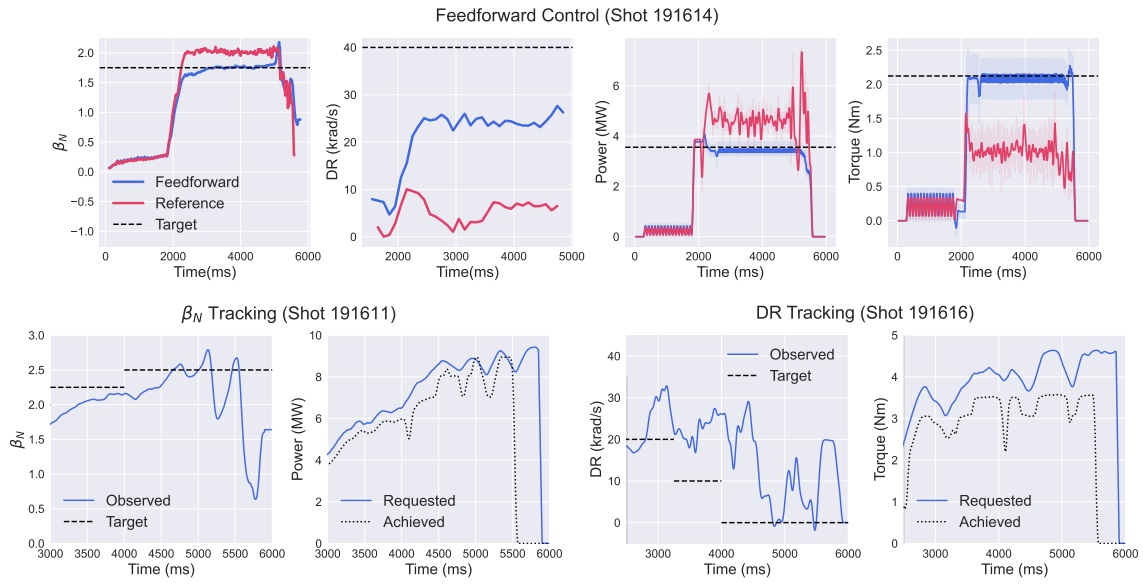

Figure 4: **Experiment Shots.** The top four plots show the β_N, DR, total power, and total torque during both shot 191614 (feedforward control) and the reference shot 164987 (red). These values are smoothed when needed and the original, unsmoothed values are shown as faded lines. The DR values are taken after doing preprocessing and dimensionality reduction via PCA. For the bottom plots, the left pair of plots shows the experiment controlling the power to hit β_N targets, while the right pair of plots shows controlling the torque injected to hit DR targets. In each pair, we show the requested amount of power or torque requested by our controller vs the actual value achieved.

DR achieved was lower than the target, one can see that it does achieve higher DR. For reference, DR has a standard deviation of 35.2 and an interquartile range of 47.2 amongst all shots in the dataset, so the error between the target and the value achieved is not as bad as it may appear.

Feedback Control Next, we tested the learned policy's ability to do feedback control. We started by having the controller track increasing β_N targets. Because β_N primarily relies on the power injected, we used the policy to control the injected power only and set the total torque to be $2\ \mathrm{Nm}$ throughout the shot. For shot 191611 in Figure 4, one can see that the controller increases the power in order to hit the target values. The last target is overshot slightly and some oscillatory behavior occurs. After the experiment, we identified a bug in our set up where the magnitude of change in power is greater than what was requested by the controller when there is high beam usage. This occurs at the 4500 ms point onward, and this phenomenon could perhaps be the reason behind the oscillatory behavior. The fluctuation in β_N is then further exacerbated by a disruption in the plasma, and all control is lost. Because of limited time, we were unable to compare against pre-existing controllers on our set up (Boyer et al., 2019; Scoville et al., 2007). While they are likely to track β_N more reliably, we still believe that this is a step in the right direction for showing MBRL's value in learning controls.

Unlike β_N tracking, there is no other controller in the DIII-D PCS that specifically tracks the difference in rotation between the $q = 1$ and $q = 2$ surfaces. To test our controller's ability to do so, we set a series of decreasing DR targets for the controller to achieve using only total torque. We set the power injected to a constant value of $5\ \mathrm{MW}$, and although torque could vary since the 210 beams were in the counter-current orientation, this still restricts the total torques that can be achieved. The

controller is unable to track the DR targets nearly as well as the β_N targets. While there are some instances of the policy doing the right thing (e.g. torque is decreased at time 4000 ms time to drop DR to the target), the policy shortly after observes DR dropping too quickly and raises torque back up again (shot 191616 in Figure 4).

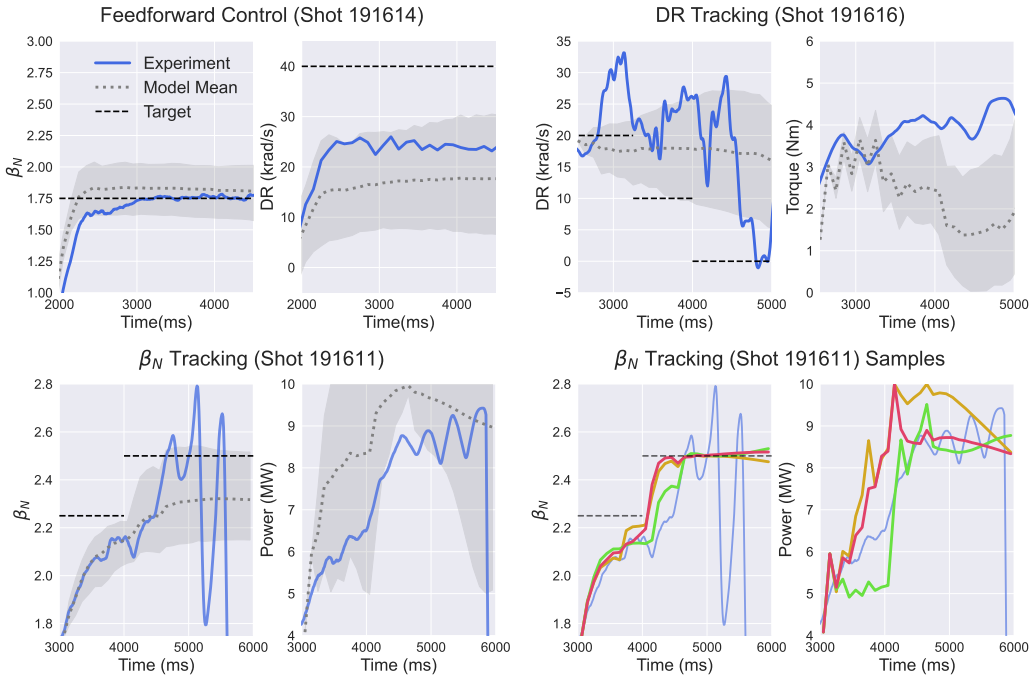

Figure 5: **Replay of Experiments.** The top and bottom left pair of plots show the experiment observations (blue) and the mean prediction from the model (gray dotted line). We used 20 sampled shots per model in the ensemble, i.e. we used 300 samples for shot 191614 and 100 samples for the other shots. The gray region is the area spanned by the 5^{th} and 95^{th} percentile sample. The bottom right pair of plots show the three highest return samples (shown in red, yellow, and green) for the β_N feedback control shot.

4.1. Post Experiment Analysis

To aid in the analysis of these experiments, we can see how predictions in our dynamics model line up with what actually happened in the experiment. Starting with the feedforward shot (191614), predictions were made using the reference shot; however instead of the original power and torque controls, the planned controls are used instead. In Figure 5, one can see that the that the true shot is indeed contained within the predicted distribution. Despite the model predicting that the target DR would not be achieved, this was the optimal configuration landed on by the model because increasing the torque injected causes β_N to overshoot the target in the simulated environment. Moreover, because the spread in DR is much higher than β_N, the optimizer implicitly favors tracking β_N.

Next, for the β_N tracking shot (191611), we replayed the shot in our simulated environment using the learned controller. Many of the sampled trajectories cannot achieve the second target of $\beta_N = 2.5$. This may be due to the fact that this is a relatively high β_N value for this reference shot, and the dynamics model may have learned that a loss of confinement usually happens at this range of β_N. When this happens, the controller keeps increasing the power in order to try to achieve

the target value. For the samples of the dynamics model that are able to achieve higher β_N, the controller is able to hit the target well, and the schedules in power injected used to achieve the targets are comparable to what was seen in the actual experiment (Figure 5). For these samples, there is no overshoot of the target or oscillation of β_N, and it is possible that without the problem with the beams that the controller would have been able to hit the target more reliably during the experiment.

Lastly we replayed the shot 191616, and use the controller to try to achieve the DR targets. Unlike control of β_N, it does not seem that DR is controllable to the same extent for this shot. However, looking at the values of the torque injected there are clear changes in the torques as the target values change. We also find that the controller does not make any drastic changes to torque if it cannot hit the DR target. We hypothesize that this is because β_N is affected by the torque injected, and it is the more reliable quantity to track. The controller will therefore not drastically change the torque if it compromises the tracking of β_N. Furthermore, we find that the uncertainty in the model increases with lower settings of torque, which may further deter the controller from decreasing torque.

5. Discussion

In this work, we show the first steps towards doing feedback control on a tokamak by learning through logged data alone. Furthermore, through our feedforward controls, we have demonstrated the predictive ability of our dynamics models for control. While we faced challenges throughout the course of this work that are common to every application of sim2real (Ibarz et al., 2021) and offline RL (Levine et al., 2020), we believe that our work provides takeaways for the RL community.

MBRL on Undirected Data. Many offline RL benchmark tasks assume that much of the collected data has the test-time task or reward function in mind. While the D4RL benchmarks (Fu et al., 2020a) do have tasks with undirected datasets (e.g. the maze tasks and FrankaKitchen), these datasets contain sub-trajectories of good behavior that simply need to be "stitched" together. We believe that our application falls into another interesting setting that these baselines do not cover. This setting is one in which there are not necessarily sub-trajectories of good behavior, but reasonable dynamics models can be learned either because the dataset is sufficiently expansive or through injecting prior information into the models (e.g. Mehta et al. (2021); Yin et al. (2021)). Unique challenges and opportunities would likely come from further studying this setting.

Model Diversity. As seen in Section 4.1, it was important to have diversity in the dynamics models to ensure that the true dynamics are covered and that the controller can handle different possibilities. While we used PPO in conjunction with these models, it is possible better results could be achieved by using recent developments that leverage the diverse model predictions for test-time adaptation (Ghosh et al., 2022; Chen et al., 2021). This also raises the question: how should one evaluate uncertainty estimates in this setting? While we chose to use miscalibration area as our metric in this work, it is unclear which metric is indicative of good policies being learned downstream.

Policy Evaluation. While it is known that models are a useful tool for off-policy evaluation (Thomas and Brunskill, 2016; Jiang and Li, 2016), we believe that it is important that these models are learned in such a way that they can test the generalization capabilities of the policy. Our method of doing this was to set aside some shots that only these testing models would train on; however, there are possibly more sophisticated procedures for doing this. This was useful for making key decisions for our learning pipeline (e.g. we found policies trained with SAC (Haarnoja et al., 2018) had worse generalization compared to those trained with PPO).

Acknowledgement

We would like to thank Nikolas Logan, Jayson Barr, and everyone at the DIII-D National Fusion Facility that helped with the preparation and running of this experiment.

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Fusion Energy Sciences, using the DIII-D National Fusion Facility, a DOE Office of Science user facility, under Awards DE-AC02-09CH1146 and DE-FC02-04ER54698. This work was also supported by DE-SC0021414 and DE- SC0021275 (Machine Learning for Real-time Fusion Plasma Behavior Prediction and Manipulation). Additionally, this work is supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE1745016 and DGE2140739. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation.

References

Joseph Abbate, R Conlin, and E Kolemen. Data-driven profile prediction for diii-d. *Nuclear Fusion*, 61(4):046027, 2021.

Joseph Abbate, Rory Conlin, Keith Erickson, and Kolemen Egemen. Data-driven control in tokamaks. *Journal of Plasma Physics (In Submission.)*, 2023.

Brandon Amos, Samuel Stanton, Denis Yarats, and Andrew Gordon Wilson. On the model-based stochastic value gradient for continuous reinforcement learning. In *Learning for Dynamics and Control*, pages 6–20. PMLR, 2021.

Gaon An, Seungyong Moon, Jang-Hyun Kim, and Hyun Oh Song. Uncertainty-based offline reinforcement learning with diversified q-ensemble. *Advances in neural information processing systems*, 34:7436–7447, 2021.

Kavosh Asadi, Dipendra Misra, and Michael Littman. Lipschitz continuity in model-based reinforcement learning. In *International Conference on Machine Learning*, pages 264–273. PMLR, 2018.

EA Baltz, E Trask, M Binderbauer, M Dikovsky, H Gota, R Mendoza, JC Platt, and PF Riley. Achievement of sustained net plasma heating in a fusion experiment with the optometrist algorithm. *Scientific reports*, 7(1):1–7, 2017.

Laszlo Bardoczi, NC Logan, and EJ Strait. Neoclassical tearing mode seeding by nonlinear three-wave interactions in tokamaks. *Physical Review Letters*, 127(5):055002, 2021.

Gregory Benton, Wesley Maddox, Sanae Lotfi, and Andrew Gordon Gordon Wilson. Loss surface simplexes for mode connecting volumes and fast ensembling. In *International Conference on Machine Learning*, pages 769–779. PMLR, 2021.

Mark D Boyer and Jason Chadwick. Prediction of electron density and pressure profile shapes on nstx-u using neural networks. *Nuclear Fusion*, 61(4):046024, 2021.

MD Boyer, KG Erickson, BA Grierson, DC Pace, JT Scoville, J Rauch, BJ Crowley, JR Ferron, SR Haskey, DA Humphreys, et al. Feedback control of stored energy and rotation with variable beam energy and perveance on diii-d. *Nuclear Fusion*, 59(7):076004, 2019.

MD Boyer, C Rea, and M Clement. Toward active disruption avoidance via real-time estimation of the safe operating region and disruption proximity in tokamaks. *Nuclear Fusion*, 62(2):026005, 2021.

RJ Buttery, RJ La Haye, P Gohil, GL Jackson, H Reimerdes, EJ Strait, and DIII-D Team. The influence of rotation on the β n threshold for the 2/ 1 neoclassical tearing mode in diii-d. *Physics of Plasmas*, 15(5):056115, 2008.

Ian Char, Youngseog Chung, Willie Neiswanger, Kirthevasan Kandasamy, Andrew O Nelson, Mark Boyer, Egemen Kolemen, and Jeff Schneider. Offline contextual bayesian optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

Xiong-Hui Chen, Yang Yu, Qingyang Li, Fan-Ming Luo, Zhiwei Qin, Wenjie Shang, and Jieping Ye. Offline model-based adaptable policy learning. *Advances in Neural Information Processing Systems*, 34:8432–8443, 2021.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. *Advances in neural information processing systems*, 31, 2018.

Youngseog Chung, Ian Char, Han Guo, Jeff Schneider, and Willie Neiswanger. Uncertainty toolbox: an open-source library for assessing, visualizing, and improving uncertainty quantification. *arXiv preprint arXiv:2109.10254*, 2021.

S Coda, M Agostini, Raffaele Albanese, S Alberti, E Alessi, S Allan, J Allcock, R Ambrosino, H Anand, Y Andrèbe, et al. Physics research on the tcv tokamak facility: from conventional to alternative scenarios and beyond. *Nuclear Fusion*, 59(11):112023, 2019.

Rory Conlin, Keith Erickson, Joseph Abbate, and Egemen Kolemen. Keras2c: A library for converting keras neural networks to real-time compatible c. *Engineering Applications of Artificial Intelligence*, 100:104182, 2021.

Jonas Degrave, Federico Felici, Jonas Buchli, Michael Neunert, Brendan Tracey, Francesco Carpanese, Timo Ewalds, Roland Hafner, Abbas Abdolmaleki, Diego de Las Casas, et al. Magnetic control of tokamak plasmas through deep reinforcement learning. *Nature*, 602(7897):414–419, 2022.

Vladimir Feinberg, Alvin Wan, Ion Stoica, Michael I Jordan, Joseph E Gonzalez, and Sergey Levine. Model-based value estimation for efficient model-free reinforcement learning. *arXiv preprint arXiv:1803.00101*, 2018.

Justin Fu, Aviral Kumar, Ofir Nachum, George Tucker, and Sergey Levine. D4rl: Datasets for deep data-driven reinforcement learning. *arXiv preprint arXiv:2004.07219*, 2020a.

Yichen Fu, David Eldon, Keith Erickson, Kornee Kleijwegt, Leonard Lupin-Jimenez, Mark D Boyer, Nick Eidietis, Nathaniel Barbour, Olivier Izacard, and Egemen Kolemen. Machine learning control for disruption and tearing mode avoidance. *Physics of Plasmas*, 27(2):022501, 2020b.

Scott Fujimoto, Herke Hoof, and David Meger. Addressing function approximation error in actor-critic methods. In *International conference on machine learning*, pages 1587–1596. PMLR, 2018.

Dibya Ghosh, Anurag Ajay, Pulkit Agrawal, and Sergey Levine. Offline rl policies should be trained to be adaptive. In *International Conference on Machine Learning*, pages 7513–7530. PMLR, 2022.

P Gohil, KH Burrell, RJ Groebner, J Kim, WC Martin, EL McKee, and RP Seraydarian. The charge exchange recombination diagnostic system on the diii-d tokamak. Technical report, General Atomics, 1991.

BA Grierson, MA Van Zeeland, JT Scoville, B Crowley, I Bykov, JM Park, WW Heidbrink, A Nagy, SR Haskey, and D Liu. Testing the diii-d co/counter off-axis neutral beam injected power and ability to balance injected torque. *Nuclear Fusion*, 61(11):116049, 2021.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

David Humphreys, G Ambrosino, Peter de Vries, Federico Felici, Sun H Kim, Gary Jackson, A Kallenbach, Egemen Kolemen, J Lister, D Moreau, et al. Novel aspects of plasma control in iter. *Physics of Plasmas*, 22(2):021806, 2015.

Julian Ibarz, Jie Tan, Chelsea Finn, Mrinal Kalakrishnan, Peter Pastor, and Sergey Levine. How to train your robot with deep reinforcement learning: lessons we have learned. *The International Journal of Robotics Research*, 40(4-5):698–721, 2021.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. *Advances in Neural Information Processing Systems*, 32, 2019.

Nan Jiang and Lihong Li. Doubly robust off-policy value evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 652–661. PMLR, 2016.

Rahul Kidambi, Aravind Rajeswaran, Praneeth Netrapalli, and Thorsten Joachims. Morel: Model-based offline reinforcement learning. *Advances in neural information processing systems*, 33: 21810–21823, 2020.

Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.

Thanard Kurutach, Ignasi Clavera, Yan Duan, Aviv Tamar, and Pieter Abbeel. Model-ensemble trust-region policy optimization. *arXiv preprint arXiv:1802.10592*, 2018.

Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643*, 2020.

Shunjie Li, H Jiang, Zhigang Ren, and C Xu. Optimal tracking for a divergent-type parabolic pde system in current profile control. In *Abstract and Applied Analysis*, volume 2014. Hindawi, 2014.

Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

James L Luxon. A design retrospective of the diii-d tokamak. *Nuclear Fusion*, 42(5):614, 2002.

M Margo, B Penaflor, H Shen, J Ferron, D Piglowski, P Nguyen, J Rauch, M Clement, A Battey, and C Rea. Current state of diii-d plasma control system. *Fusion Engineering and Design*, 150: 111368, 2020.

Dale Meade. 50 years of fusion research. *Nuclear Fusion*, 50(1):014004, 2009.

Viraj Mehta, Ian Char, Willie Neiswanger, Youngseog Chung, Andrew Nelson, Mark Boyer, Egemen Kolemen, and Jeff Schneider. Neural dynamical systems: Balancing structure and flexibility in physical prediction. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 3735–3742. IEEE, 2021.

Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016.

Matthew S Parsons. Interpretation of machine-learning-based disruption models for plasma control. *Plasma Physics and Controlled Fusion*, 59(8):085001, 2017.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

PA Politzer, CC Petty, RJ Jayakumar, TC Luce, MR Wade, JC DeBoo, JR Ferron, P Gohil, CT Holcomb, AW Hyatt, et al. Influence of toroidal rotation on transport and stability in hybrid scenario plasmas in diii-d. *Nuclear Fusion*, 48(7):075001, 2008.

Christina Rea, KJ Montes, KG Erickson, RS Granetz, and RA Tinguely. A real-time machine learning-based disruption predictor in diii-d. *Nuclear Fusion*, 59(9):096016, 2019.

H Reimerdes, AM Garofalo, GL Jackson, M Okabayashi, EJ Strait, Ming-Sheng Chu, Y In, RJ La Haye, MJ Lanctot, YQ Liu, et al. Reduced critical rotation for resistive-wall mode stabilization in a near-axisymmetric configuration. *Physical review letters*, 98(5):055001, 2007.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pages 1889–1897. PMLR, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

JT Scoville, DA Humphreys, JR Ferron, and P Gohil. Simultaneous feedback control of plasma rotation and stored energy on the diii-d tokamak. *Fusion engineering and design*, 82(5-14): 1045–1050, 2007.

J Seo, Y-S Na, B Kim, CY Lee, MS Park, SJ Park, and YH Lee. Development of an operation trajectory design algorithm for control of multiple 0d parameters using deep reinforcement learning in kstar. *Nuclear Fusion*, 62(8):086049, 2022.

Jaemin Seo, Y-S Na, B Kim, CY Lee, MS Park, SJ Park, and YH Lee. Feedforward beta control in the kstar tokamak by deep reinforcement learning. *Nuclear Fusion*, 61(10):106010, 2021.

Ricardo Shousha, Keith Erickson, and Egemen Kolemen. Development and experimental demonstration of real-time kinetic profile fitting algorithm for improved equilibrium reconstruction (rtefit) and control on diii-d. *Journal of Plasma Physics (In Submission.)*, 2023.

Philip Thomas and Emma Brunskill. Data-efficient off-policy policy evaluation for reinforcement learning. In *International Conference on Machine Learning*, pages 2139–2148. PMLR, 2016.

B Tobias, M Chen, IGJ Classen, CW Domier, R Fitzpatrick, BA Grierson, NC Luhmann Jr, CM Muscatello, M Okabayashi, KEJ Olofsson, et al. Rotation profile flattening and toroidal flow shear reversal due to the coupling of magnetic islands in tokamaks. *Physics of Plasmas*, 23(5):056107, 2016.

Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

T Wakatsuki, T Suzuki, N Oyama, and N Hayashi. Ion temperature gradient control using reinforcement learning technique. *Nuclear Fusion*, 61(4):046036, 2021.

Michael L Walker, Peter De Vries, Federico Felici, and Eugenio Schuster. Introduction to tokamak plasma control. In *2020 American Control Conference (ACC)*, pages 2901–2918. IEEE, 2020.

ML Walker, DA Humphreys, and JR Ferron. Multivariable shape control development on the diii-d tokamak. In *17th IEEE/NPSS Symposium Fusion Engineering (Cat. No. 97CH36131)*, volume 1, pages 556–559. IEEE, 1997.

Mitchell Wortsman, Maxwell C Horton, Carlos Guestrin, Ali Farhadi, and Mohammad Rastegari. Learning neural network subspaces. In *International Conference on Machine Learning*, pages 11217–11227. PMLR, 2021.

Yifan Wu, George Tucker, and Ofir Nachum. Behavior regularized offline reinforcement learning. *arXiv preprint arXiv:1911.11361*, 2019.

Yuan Yin, Vincent Le Guen, Jérémie Dona, Emmanuel de Bézenac, Ibrahim Ayed, Nicolas Thome, and Patrick Gallinari. Augmenting physical models with deep networks for complex dynamics forecasting. *Journal of Statistical Mechanics: Theory and Experiment*, 2021(12):124012, 2021.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Y Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. Mopo: Model-based offline policy optimization. *Advances in Neural Information Processing Systems*, 33:14129–14142, 2020.

Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. Combo: Conservative offline model-based policy optimization. *Advances in neural information processing systems*, 34:28954–28967, 2021.