ML in Slow Spill Regulation for Mu2e

Aakaash Narayanan

4th Slow Extraction Workshop
26 January 2022
Mu2e and the need for Proton Pulses

Mu2e is an upcoming experiment at Fermilab that intends to capture muons in Aluminum atoms and look for new physics in their decay to electrons.

To create the muons, proton pulses are made to hit a production target and muons are selected from the secondaries.

To help facilitate the signal strength, Mu2e demands pulses of muons arrive at the Aluminum target with strict requirements on the rate uniformity.

The proton pulses with the required time structure are created by slow extraction of the bunched beam from the Delivery Ring (DR).

![Diagram of Mu2e and proton pulses](image-url)
Beam Delivery for Mu2e

The extraction from the DR is achieved by the aid of six sextupoles and a circuit of three tune quadrupoles in the DR.

The sextupoles introduce the third integer resonance strength and the fast-quadrupoles drives the beam’s tune closer to third integer resonance.

About $10^{12}$ protons would be injected in the Delivery Ring and $3 \times 10^7$ protons would be sent to the Mu2e experiment target at every revolution.

This is achieved through 29/3 resonant extraction.

<table>
<thead>
<tr>
<th>Beam Parameters for Resonant Extraction</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum Intensity in DR</td>
</tr>
<tr>
<td>Revolution Time in DR</td>
</tr>
<tr>
<td>Orbit length of DR</td>
</tr>
<tr>
<td>Horizontal tune in DR</td>
</tr>
<tr>
<td>Protons per extracted pulse</td>
</tr>
<tr>
<td>Spill Duty Factor</td>
</tr>
<tr>
<td>Single spill duration</td>
</tr>
<tr>
<td>Beam Power</td>
</tr>
</tbody>
</table>
Need for Regulation

Main types of spill non-uniformities:

- Spill underlying profile
- Fast random ripples (noise)
  ✓ Fast PID feedback regulation
- Spill coherent ripples

Spill ripples negatively impact the data:

- Detector pile-up
- Reconstruction inefficiency
- Dead time

Long high spikes are especially bad

- Regulation will be targeted at large spikes
Spill Regulation System

The slow regulation controller will be tracking the slow changes in the spill profile producing corrections to the tune quadrupole current ramp to achieve the uniform spill rate.

The fast regulation system would be supplemented on top of the slow regulation in order to correct for instantaneous ripples in the spill intensity.

The SRS would also be dealing with the n*60 Hz harmonic noise content arising from the power supplies. The controller will determine the harmonic content of the ripple and apply feedforward corrections.
Fast Regulation Loop

The fast regulation loop deals with instantaneous noises that affect the spill quality within one spill.

The instantaneous fluctuations within one spill due to random noises can be large. In the SRS, this is handled by the fast PID loop controller.

We assume here that this noise (ripples) have a random nature or otherwise are a semi-random component of regular harmonic noise that the harmonic controller is not able to suppress.

The correction signal to counter the noise will superimpose on top of the ideal tune ramp provided by the slow regulation.

The three main parameters to control the spill quality in the fast regulation loop are the three gain values of the PID controller: \( (G_P, G_I, G_D) \)
Using Machine Learning

We explore using machine learning (ML) algorithms to tune the PID gain values to achieve the improved spill quality.

The spill quality is quantified by ‘Spill Duty Factor’ (SDF), defined by:

\[
SDF = \frac{1}{1 + \sigma^2_{\text{ext. rate}}}
\]

where \(\sigma^2_{\text{ext. rate}}\) is the variance in the extraction rate computed for one full spill, assuming the average intensity is normalized to 1.

We use a simplified semi-analytical model to simulate the physics of resonant extraction in order to generate training data for the machine learning algorithm.
Physics Simulator Code

1. NOISE GENERATION
2. PID RESPONSE
3. BUTTERWORTH LOW PASS FILTER
4. TRANSIT DELAY
5. CORRECTED QUADRUPOLE CURRENT
6. BEAM EXTRACTION
7. SPILL MONITOR
Physics Simulator Blocks

1. **NOISE GENERATION**

The noise for one full spill is generated in terms of extraction rate. We assume log-normal distribution for the noise spectrum.

Since the source of noise could emanate from any of the elements in the ring, the noise is pre-generated before the spill and is added directly as fluctuations in the spill.

Ideal spill is normalized to an expectation value of 1.

2. **PID RESPONSE**

With the full extraction rate known, the PID calculates the error at every time step and computes the control signal.

At every time step, the difference between the ideal spill and the actual spill is computed, and the PID calculates the control signal to be given to counter the noise in the spill rate.
The PID response is passed through a low-pass filter to simulate the steel beam pipe suppressing any magnetic field variations greater than 1 kHz.

Once the particle jumps out of the separatrix, it does not get immediately extracted as it takes some finite amount of turns to get to the septum location.

Transit time studies were done in the tracking simulations to determine the number of turns particles take to get extracted. This delay was then modelled into the physics simulator.
The ideal tune current ramp is taken to be a logarithmic function of time in our analytical model. The delayed and low-pass filtered PID response is superimposed with the idealized logarithmic quadrupole current curve.

With the corrected current ramp, total beam extraction is computed from $t = 0$ to 43 ms. This would be the ‘fast regulated’ spill.

The spill monitor block computes the spill rate for one full spill at a time step of 10 kHz (which is the total Gain Bandwidth of the SRS). We assume the spill monitor to be fast enough to not affect the loop.
PID Tuning Scheme Using ML Simulator

The ML simulator uses the physics simulator iteratively to compute loss functions and optimize the gain values.

In the very first iteration, the neural network assigns random gain values and calls the physics simulator. The physics simulator then outputs the PID loop regulated spill rate, from which the spill duty factor is calculated.

Once the SDF is calculated, a loss function $l$ is defined to train the ML model:

$$l = (1 - SDF)^2$$

The loss value, along with the previous three gain values, are fed into the neural network.

The neural network then calculates the gradient of the loss function with respect to the PID gain (i.e., $\delta l/\delta G$) and backpropagates to update the weights in the direction of minimization of the loss function, outputting new gain values.
PID Tuning Scheme Using ML

With the updated gain values, the physics simulator is run again for a full spill, but this time with a completely new random noise profile.

After the 2\textsuperscript{nd} full spill, the SDF and the loss function are again computed and fed into the neural network to compute the loss gradients.

The neural network again updates the weights and gives a new set of \((G_P, G_I, G_D)\), and the physics simulator is called again.

This is done iteratively until the loss function becomes minimal (i.e., the PID loop’s performance becomes maximal).

We refer to this approach as a Hybrid ML Simulator because only those functions which must be differentiable (i.e., \((\delta I/\delta G)\) computable) are made so. This allows functions such as noise generation and tune ramps to be pre-computed and excluded from the more computationally expensive gradient calculation and backpropagation steps.
ML Optimization at work…

Evolution of PID gain values

Evolution of the SDF
Evolution of PID gains

Published in IPAC 2021
THPAB243
Spill Regulation ML Model

We next explore the possibility of Machine Learning entirely replacing the PID controller instead of simply tuning the gains of the PID controller.

Since the spill is temporally sensitive, a Recurrent Neural Network was chosen to train the model to emulate the PID controller.

RNNs typically suffer from ‘short-term’ memory (the ‘vanishing gradient problem’).

To overcome that, LSTM is a type of neural network that has ‘internal loops’ that enables connecting temporally sensitive past information to perform present tasks.

Image source: https://colah.github.io/posts/2015-08-Understanding-LSTMs/
LSTM Model Emulating PID

ML Regulator - Single Time Step

ML Model

Corrected Spill Values 0 through $t_n$

Corrected Spill Values 0 through $t_{n+1}$

Quadrupole current correction for $t_{n+1}$
given: ........

Corrected spill value for $t_{n+1}$
given: - + ........

Previous observations → neural network → quad current correction → differentiable simulator → next observation
LSTM Model Matching PID Performance!

Raw SDF

\[ SDF = \frac{1}{1 + \sigma^2_{\text{ext. rate}}} \]

Relative performance to PID

Blue line = \[ \frac{ML_{\text{SDF}} - Noise_{\text{SDF}}}{PID_{\text{SDF}} - Noise_{\text{SDF}}} \]
Reinforcement Learning

Reinforcement Learning (RL) is a type of ML technique that enables an agent to interact with the environment by trial and error using feedback from its own actions and experiences.

Two types of RL – model free and model based.

In our work, we investigate a model free RL, with the agent being the knob controlling quadrupole current magnitude and the environment being the process of slow extraction.
The action function \( a \) is defined as the set of actions taken by our agent in its environment.

The state function \( s \) is defined as the set of environment parameters that affect the course of our agent.

The reward function \( r \) is defined as a value we assign to a specific action \( a \) taken in a specific state \( s \).

As the agent progresses in time, we accumulate a set of \((a, s, r)\): \((s_0, a_0, r_1), (s_1, a_1, r_2), \ldots, (s_i, a_i, r_{i+1}), \ldots, (s_T, a_T, r_{T+1})\)

In the case of slow spill, the action space is continuous as the control signal’s magnitude could be any real number. To deal with continuous action space, we use policy-based actor-critic methods whereby two neural networks are trained.
Reinforcement Learning – Actor-Critic method

Two neural networks are simultaneously trained: The actor and the critic.

The actor network takes in the state space variables as the input and outputs the action (i.e., the control signal) to be superimposed to the tune ramp quad current. This action is played for the next time step and the new spill rate is obtained.

The critic network takes in the reward values for the episode and gives out a value ‘criticizing’ the how good the action taken was. The ‘criticism’ is fed back into the actor network so it takes a better action the next time.

With every iteration, both the actor and the critic network directly learns the ideal policy required to regulate the spill rate.
Work in progress - Reinforcement Learning
THANK YOU!
Hardware for the SRS

The SRS system architecture will consist of the System-On-Module (SoM) and a carrier board.

The SoM is a FPGA mezzanine card that hosts the Intel Arria10 SoC, which features a second-generation dual-core ARM Cortex-A9 MPCore processor-based hard processor system (HPS).

The FPGA primarily is intended to task itself in the ‘fast regulation’. The HPS could take care of the ‘slow regulation’ and the ‘harmonic content tracker’.