CLASSE
Cornell Laboratory for Accelerator-based ScienceS & Education

The Center for BRIGHT BEAMS
A National Science Foundation Science & Technology Center

Brookhaven National Laboratory

U.S. DEPARTMENT OF ENERGY

Exploring Machine Learning Techniques to Improve Accelerator Operation at BNL

Lucy Lin

Advisor: Georg Hoffstaetter

Yichao Jing, Kai Shih (CeC)

Bohong Huang, Weining Dai, Vincent Shoefer, Kevin Brown (AGS)

@BrookhavenLab

9/15/2022

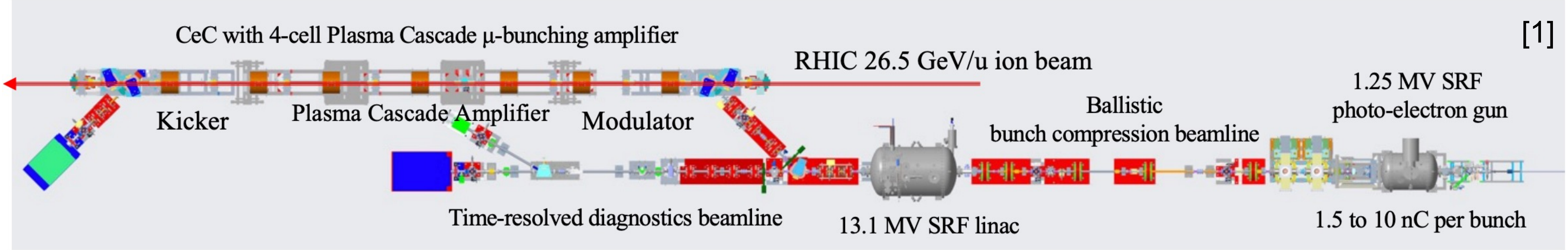
Summary

- Machine Learning for improving Coherent electron Cooling (CeC) operations

- Machine Learning for brightness control at the Alternating Gradient Synchrotron (AGS)

Coherent electron Cooling

- Designed to cool 26.5 GeV/u ion beam circulating in RHIC's yellow ring.

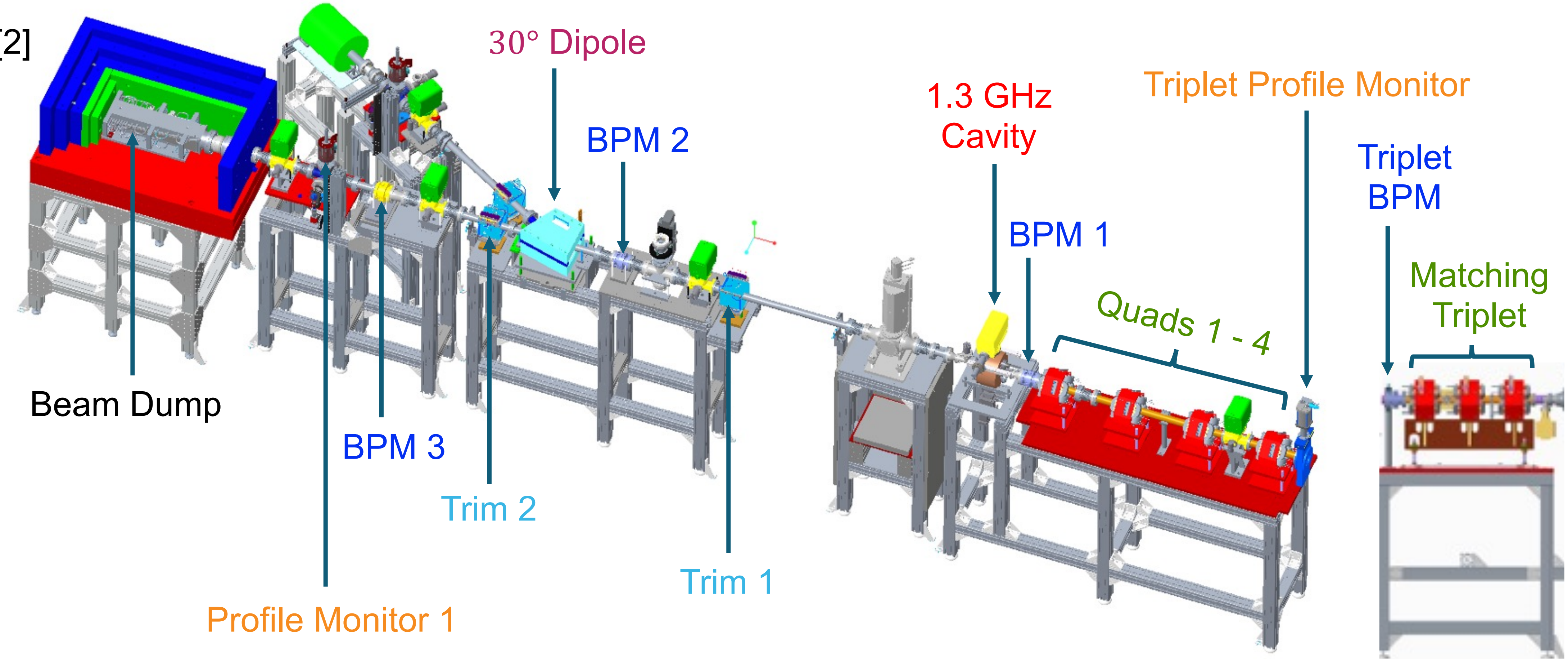


- CeC CW SRF accelerator with unique SRF electron gun generates electron beams with quality sufficient for the current experiment and for the future EIC cooler.
- Electron bunches are compressed to peak current of 50 – 100 A and accelerated to 14.5 MeV.
- Accelerated electron beam is transported through an achromatic dogleg to merge with ion beam in RHIC.
- Interaction between ions and electron beam occurs in the common section.

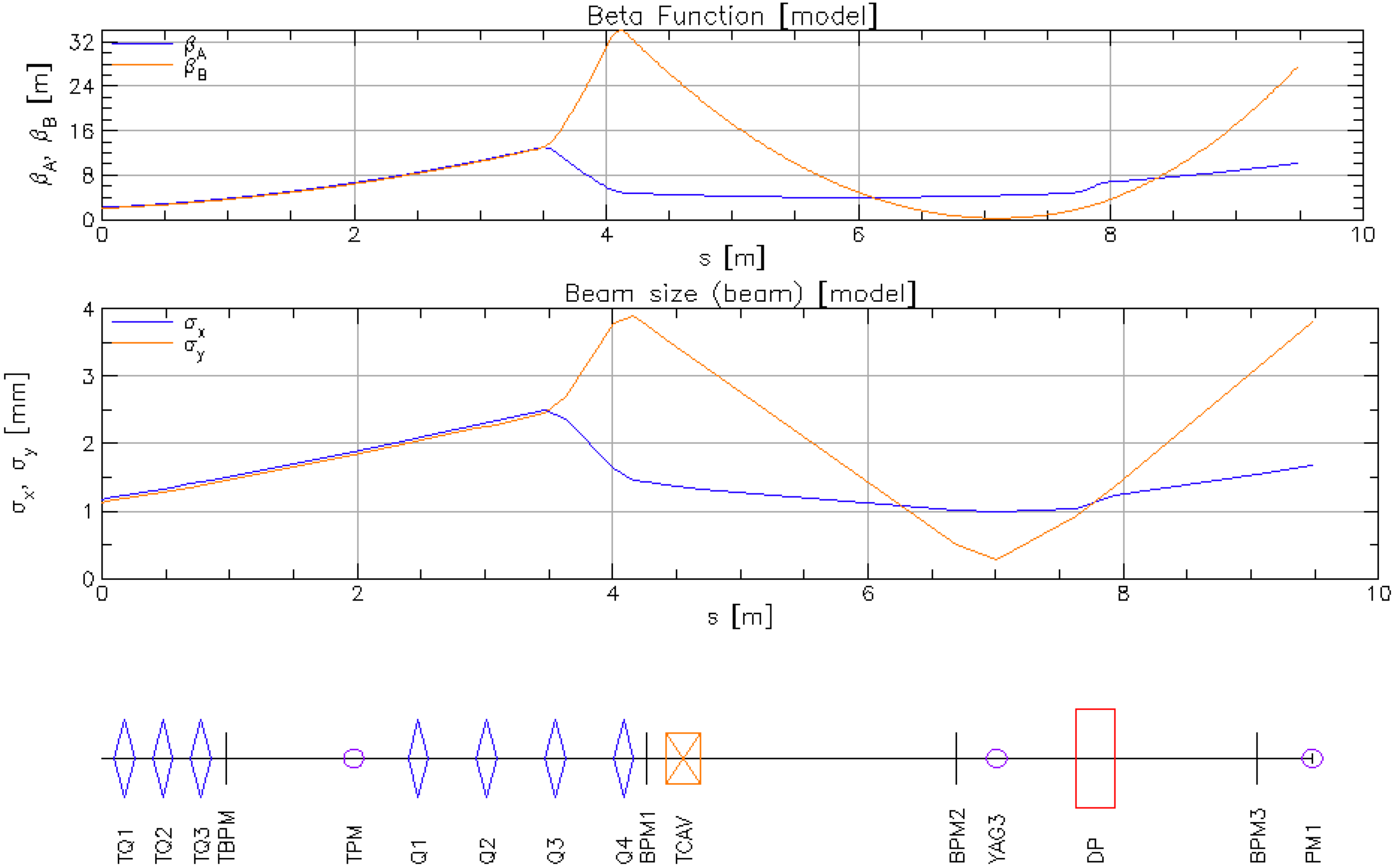
Time-resolved Diagnostic Beamline (TRDBL)

Beam line: 7 quadrupoles (3 + 4), 2 trims, 1 transverse deflecting cavity, 1 dipole
Monitors: 2 Profile Monitors, 4 BPMs

[2]

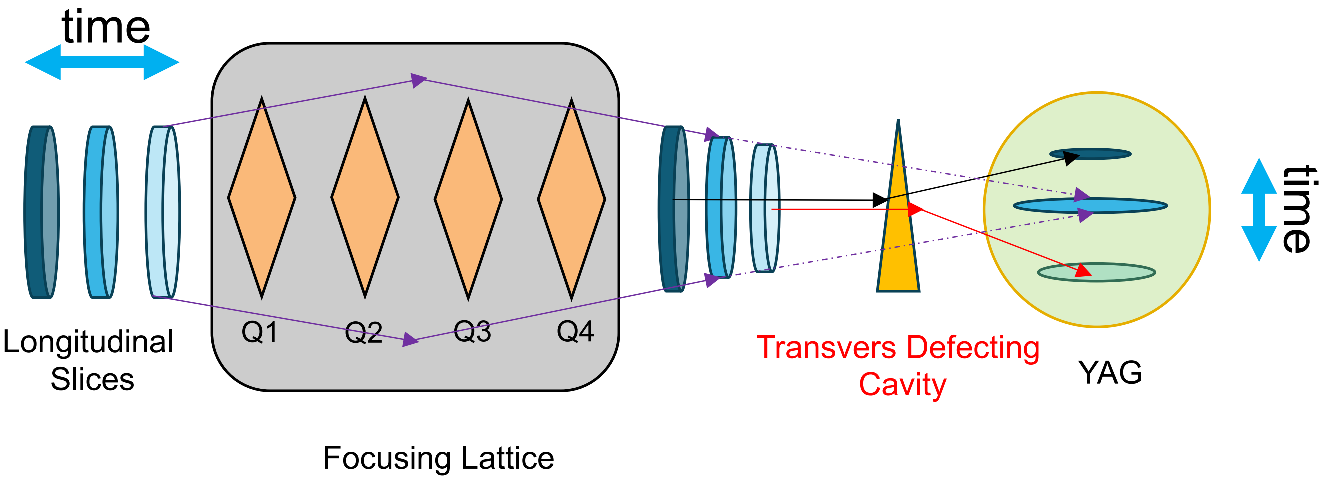


Bmad simulation of TRDBL



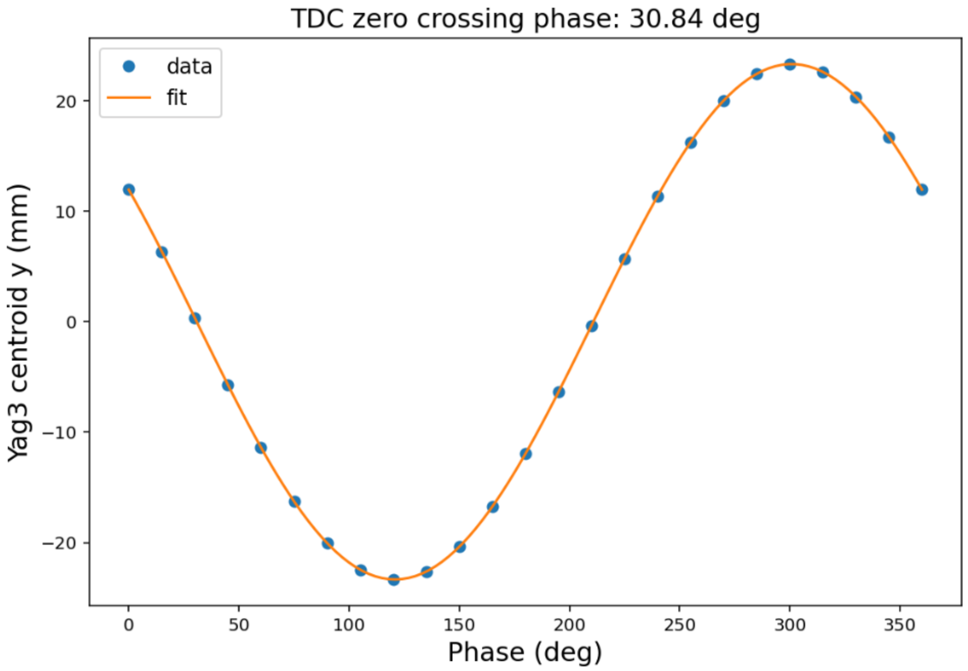
Transverse deflecting cavity (TDC)

- A TDC converts the beam’s longitudinal distribution to transverse distribution which is measurable



[3]

Phase TDC to zero crossing phase

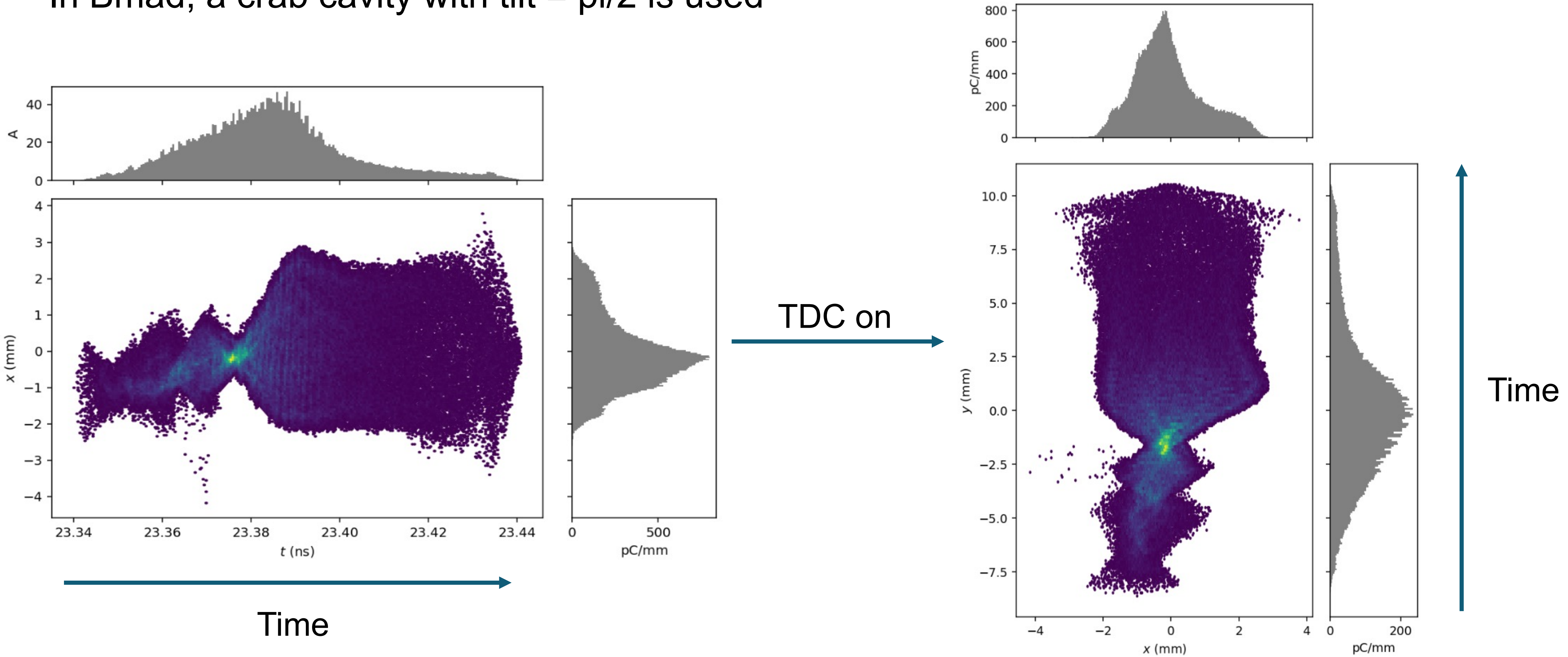


Parameter	Value
TDC RF frequency	1.3 GHz
TDC RF voltage	100 – 140 kV
Bunch length	~ 70 ps
Beam energy	14.56 MeV
Beam size σ_y at Yag without TDC	~ 0.2 – 0.4 mm

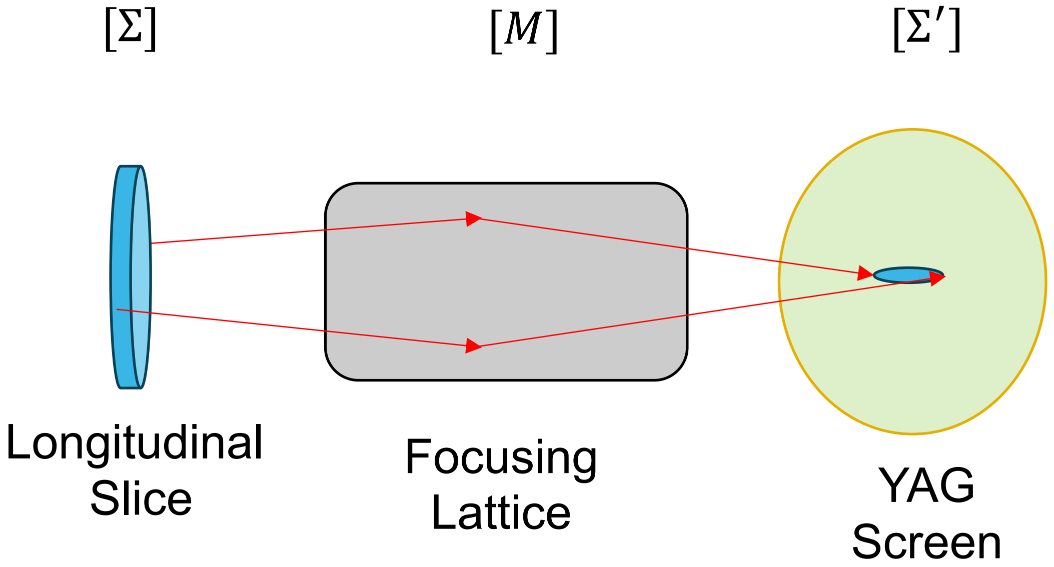
TDC simulation results: time profile

- TDC provide a time dependent transverse kick to the beam
- After TDC, the beam's time information convert to Y direction
- In Bmad, a crab cavity with tilt = pi/2 is used

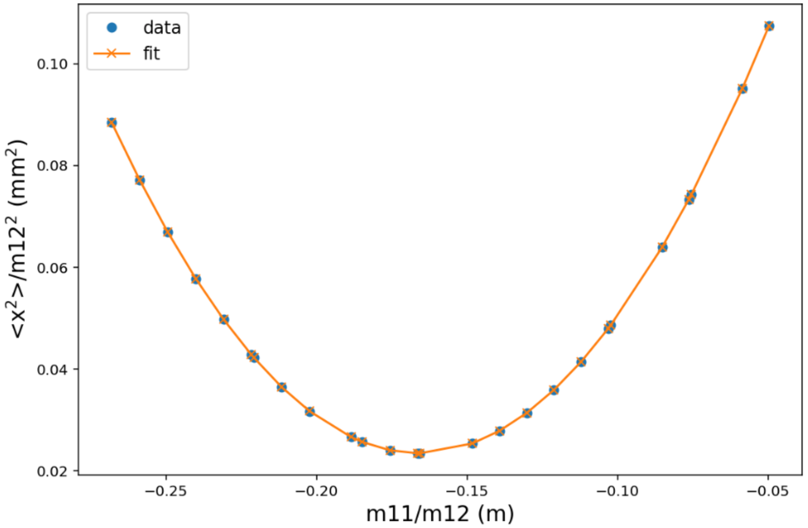
[6]



Emittance measurement



[3]



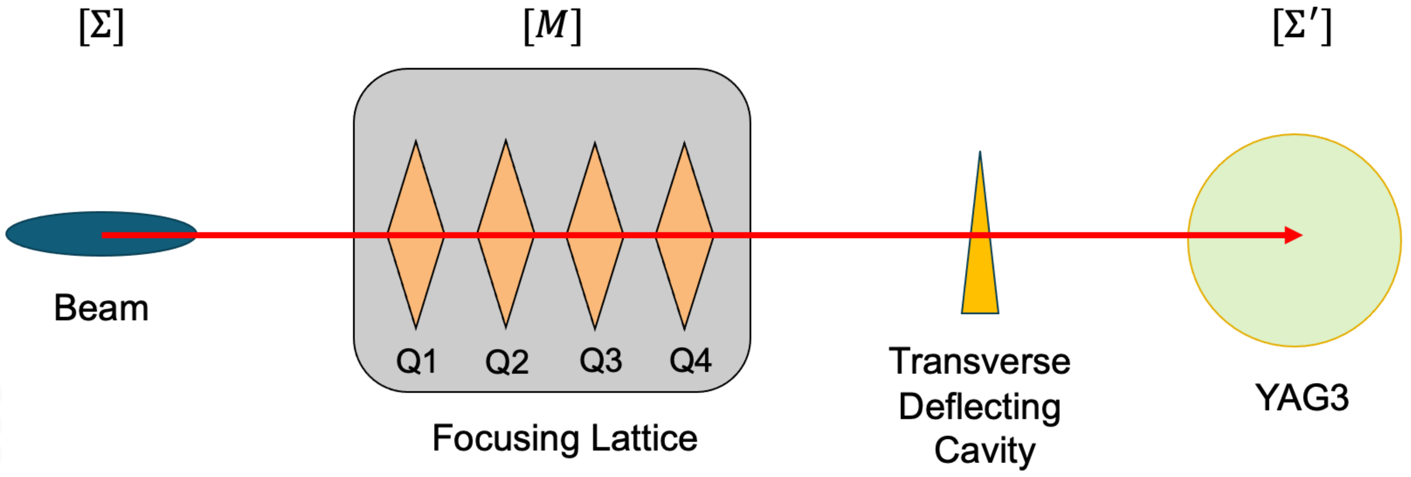
$$\because \ [\Sigma'] = [M][\Sigma][M]^T$$

$$\therefore \sigma'_{11} = m_{11}^2\sigma_{11} + m_{11}m_{12}2\sigma_{12} + m_{12}^2\sigma_{22}$$

$$\frac{\sigma'_{11}}{m_{12}^2} = \sigma_{11}\left(\frac{m_{11}}{m_{12}}\right)^2 + 2\sigma_{12}\left(\frac{m_{11}}{m_{12}}\right) + \sigma_{22}$$

$$\text{parabola fit} \Longrightarrow \varepsilon = \sqrt{\sigma_{11}\sigma_{22} - \sigma_{12}^2}$$

Quadrupole scan with two quads



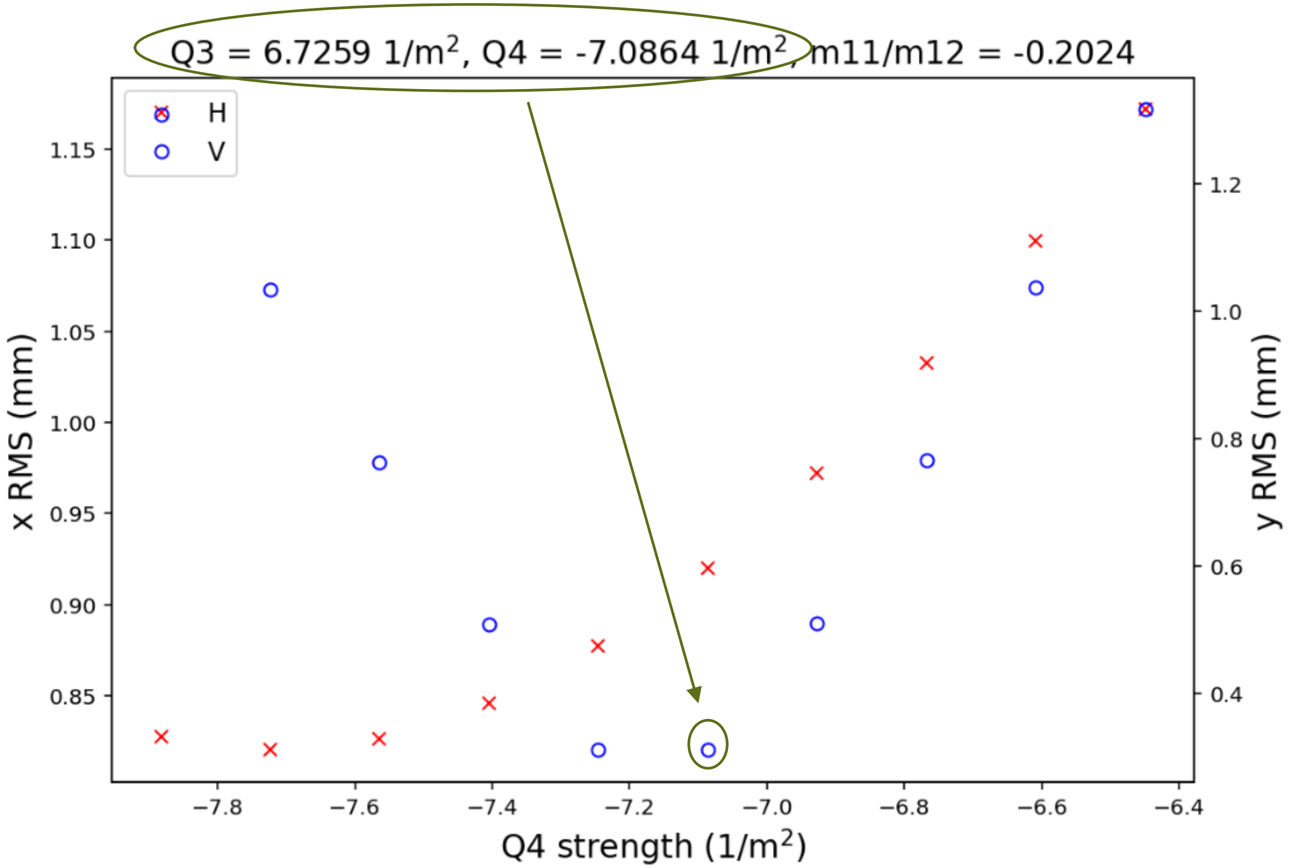
[3]

- Quad scan method with 1 quad → defocusing in another plane
- Vertical focusing → slice beam vertically to get slice emittance
- Scan two quads (Q3, Q4) with opposite polarity → keep beam focused vertically
- Find quad combination settings that gives best vertical focusing

Quadrupole scan with two quads

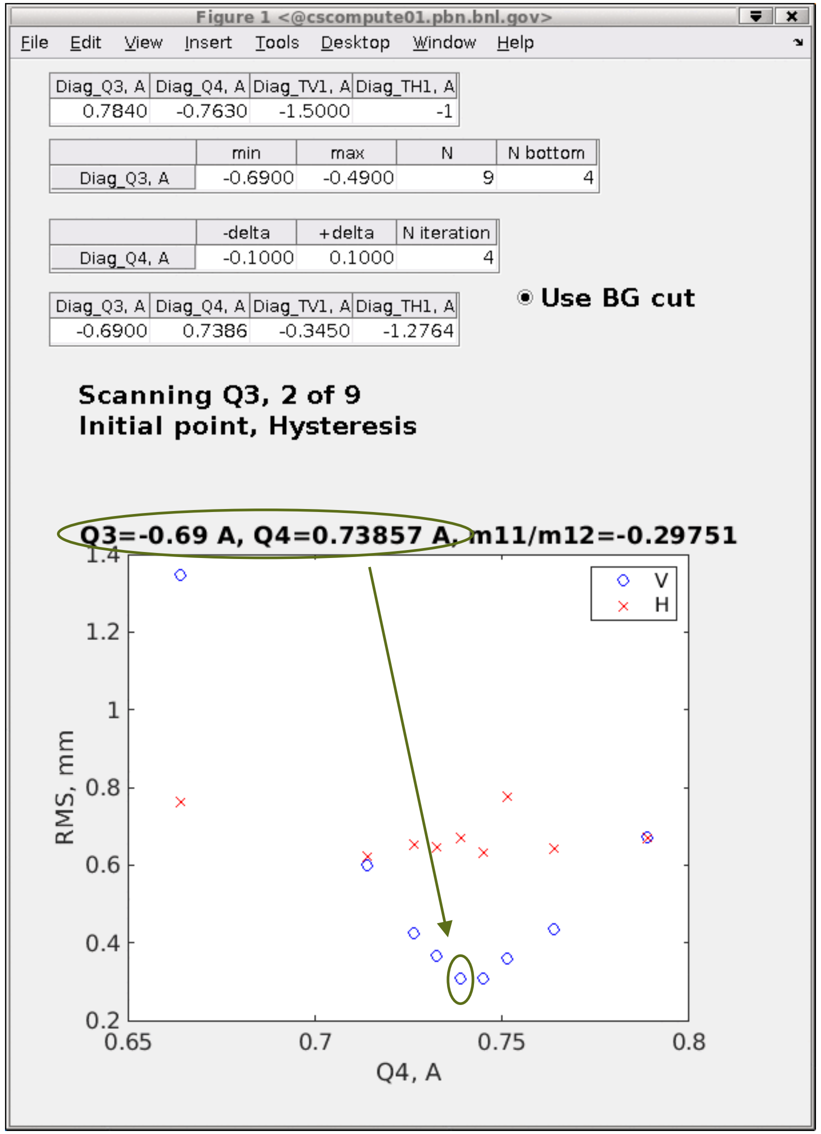
- Scan diagnostic Q3 and Q4 together, observe beam at Yag3
- For each Q3 value, find Q4 value that gives best vertical focusing at Yag3

$$[M] = [M_{Q4\ to\ YAG3}][M_{Q4}][M_{Q3\ to\ Q4}][M_{Q3}]$$



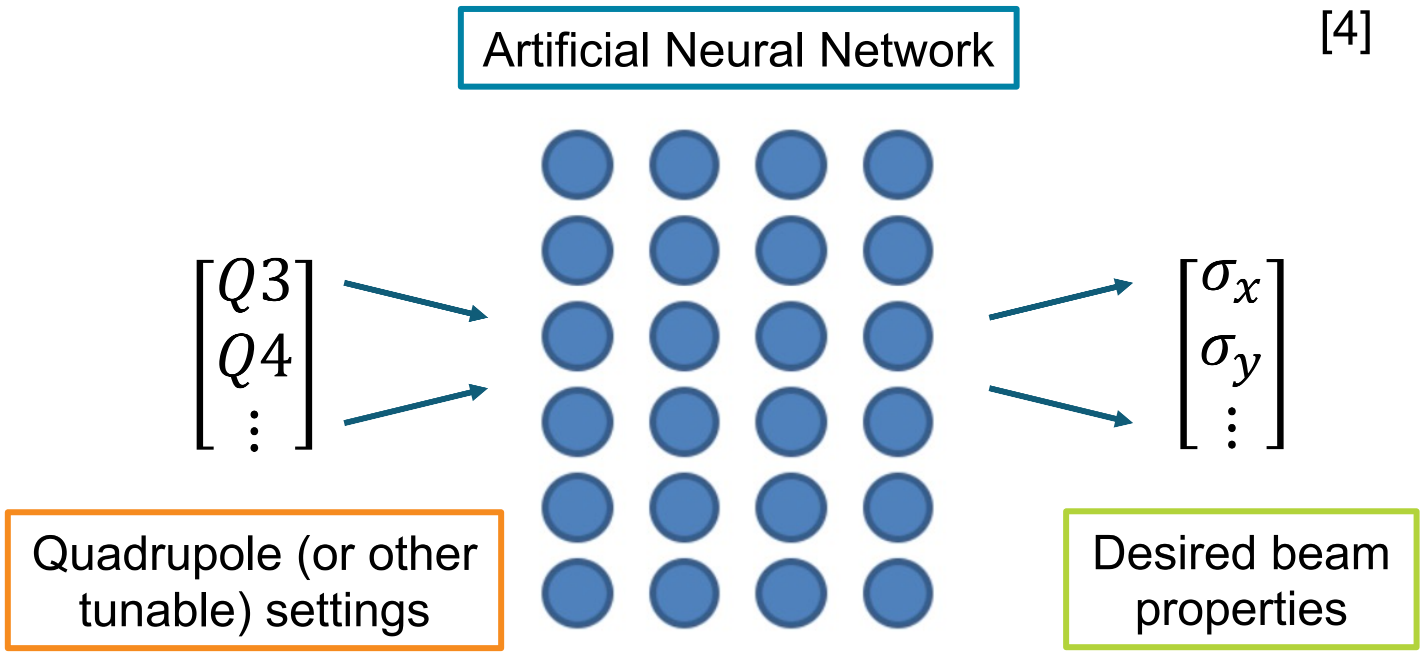
Current quad scan routine

- Find best Q3-Q4 combinations with sequential scans:
  - Scan 13 Q3 settings
  - For each Q3 setting, scan 9 Q4 settings
  - Record Q3-Q4 combination that gives smallest Y RMS
  - Calculate and store m_{11} and m_{12} for parabola fitting
- Time taken:
  - ~ 5 minutes for each Q3 setting
  - > 1 hour for an entire scan routine



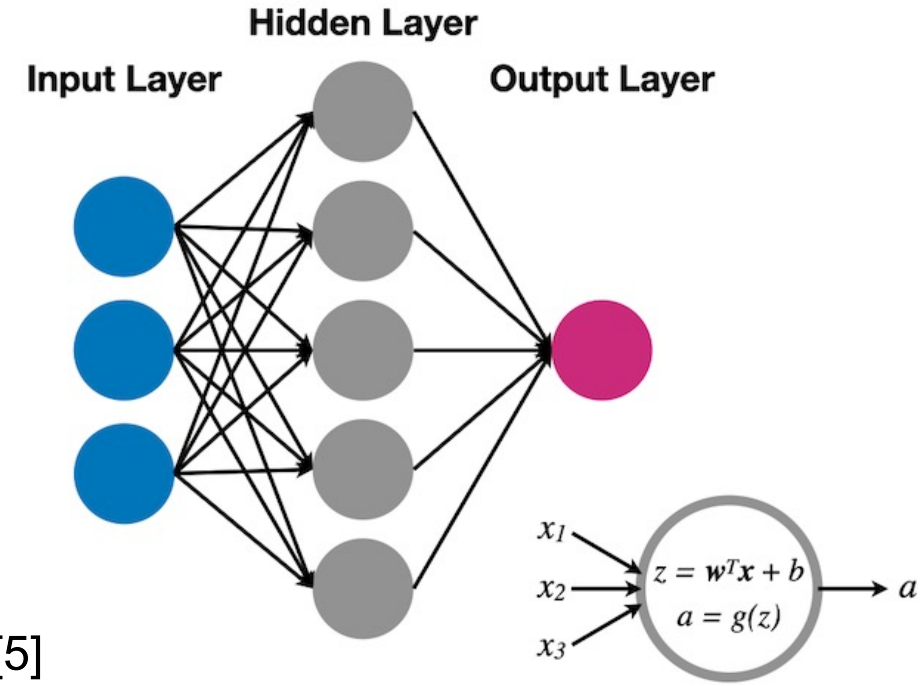
Speed up quad scan with ML

- Time consuming sequential scans
- Train a ML model to establish mapping between quadrupole settings and beam size
- Trained ML model predicts best Q3-Q4 combinations without additional scans
- Useful for faster general beam tuning & as starting point of optimization

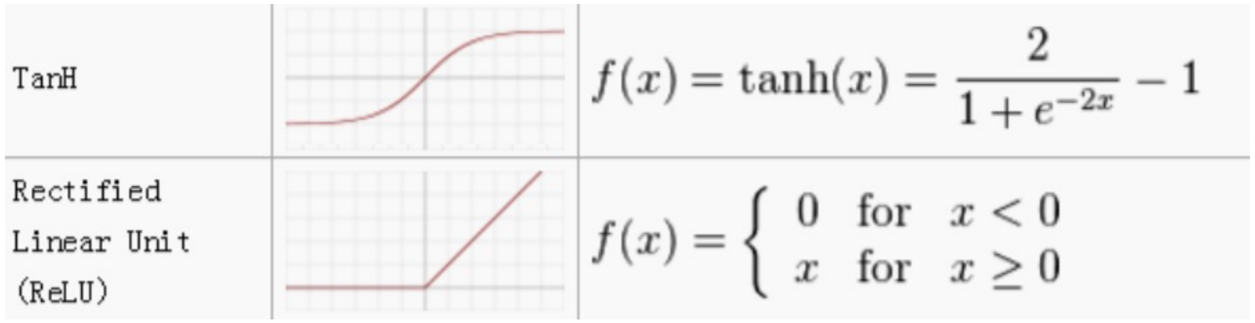


Method

- Neural Network (NN) using pytorch → USPAS course: Optimization and Machine Learning for Accelerators
- Fully connected layers: dense layer
  - output = activation(dot(input, kernel) + bias)
- Activation function: Hyperbolic Tangent (Tanh) and Rectified Linear Unit (ReLU)

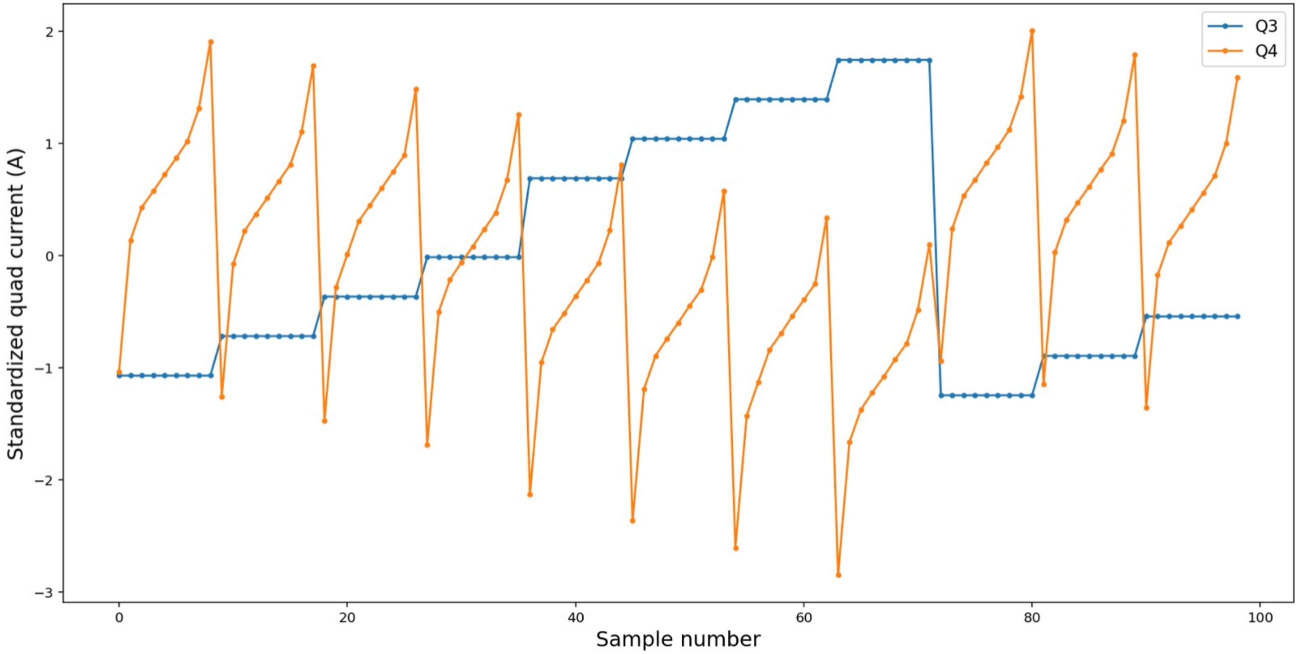


[5]

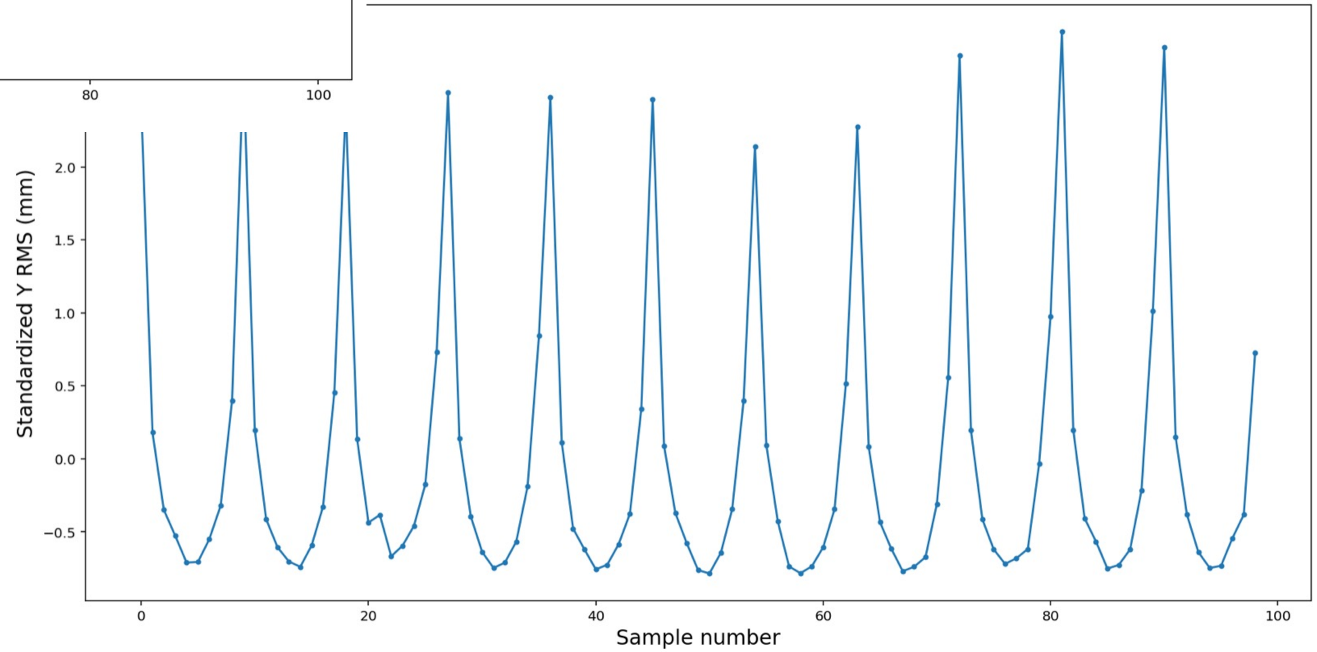


[6]

Sample historical data

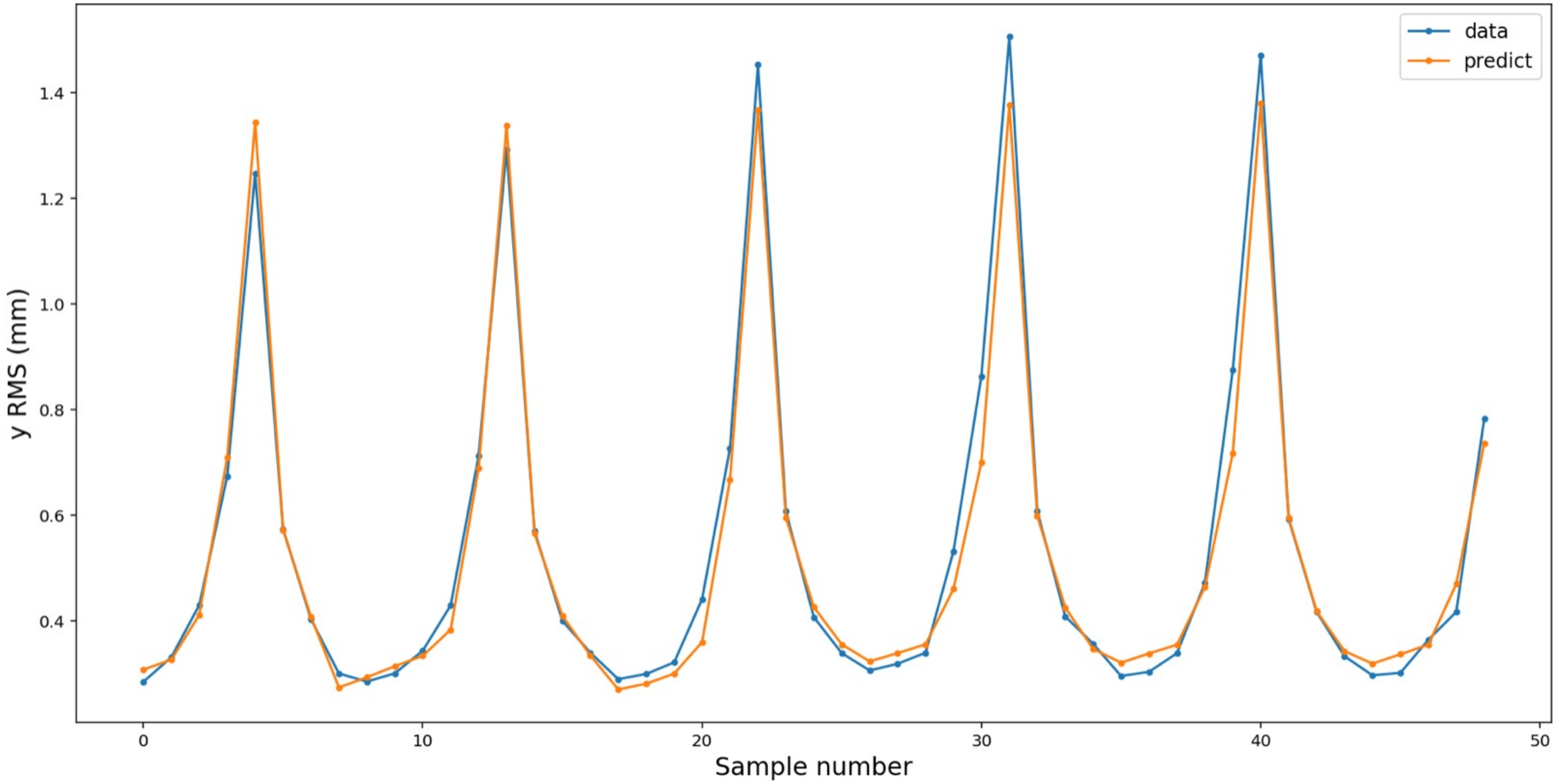


- Input: Q3, Q4 power current
- Output: RMS beam size, focus on y direction



Quadscan NN model: training results

Training: 50 out of 99 data pairs, testing (shown below): 49 out of 99 data pairs



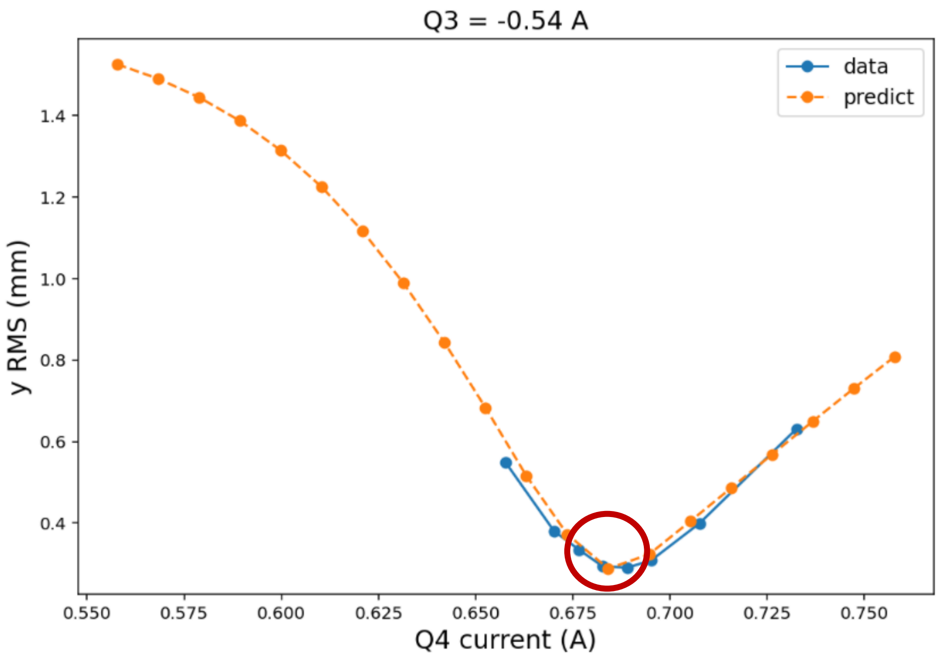
New quadscan routine with Neural Network

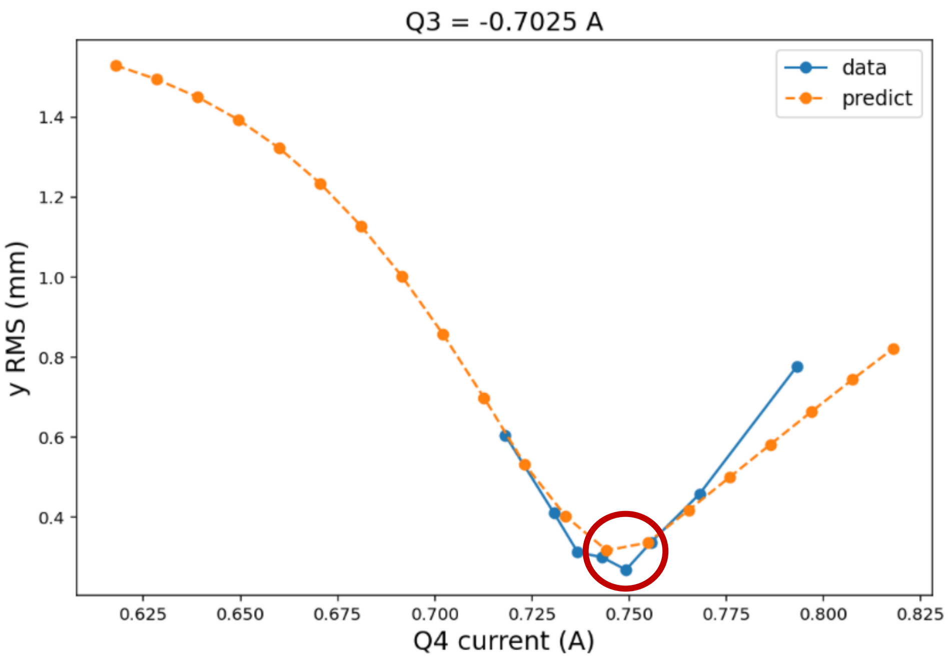
1. Scan 6 Q3 settings with old quad scan routine, but save all 54 data points
2. Train neural network model on 54 data points
3. Give neural network the remaining 7 Q3 settings needed to be scanned
4. Let neural network predict the corresponding Q4 settings that give best focusing
5. Load the predicted settings to the beamline and record beam size

- Current method:
  - Matlab script with GUI to do scan → jupyter notebook to train model and generate predicted settings → Matlab script to change beamline
- Optimal future method:
  - Incorporate everything into an executable with GUI, no need to switch codes

New quadscan routine: real historical data

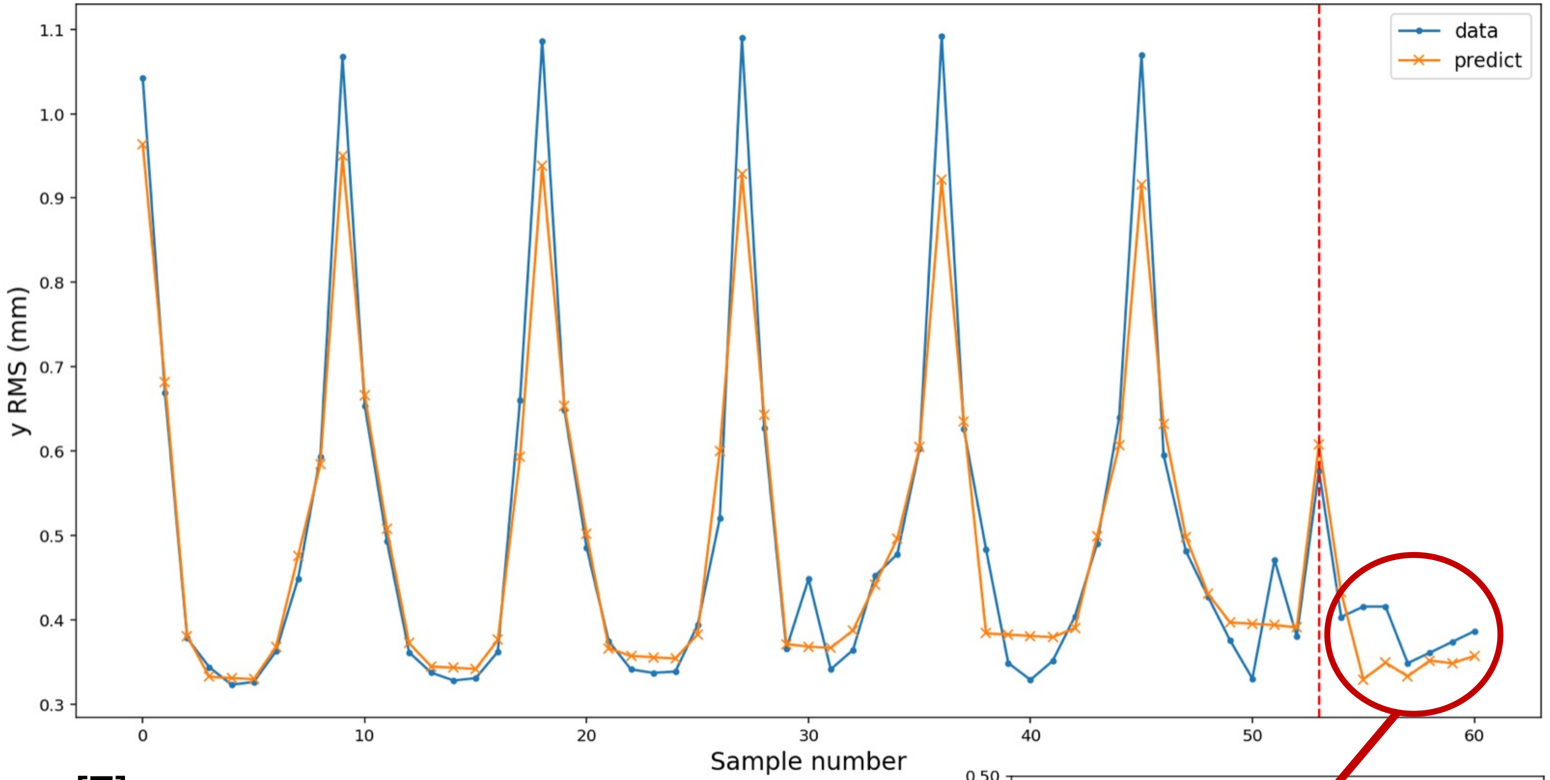
- Function findq4(q3current) from old routine gives a rough estimate of medium Q4 value
- Scan through $\Delta Q4 = \pm 0.1$ around the medium value with trained NN
- Pick the Q4 value that gives smallest predicted Y RMS value
- Compare to actual Y RMS value





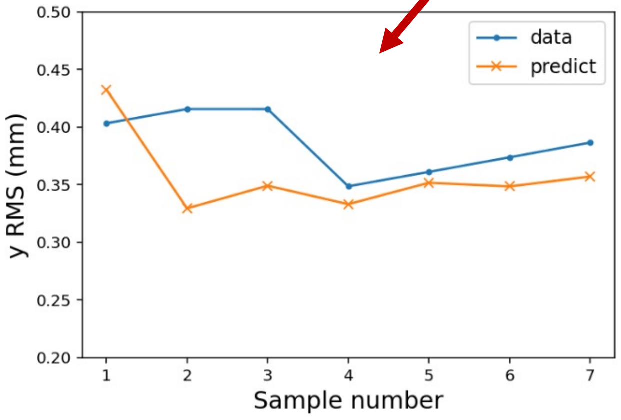
Test new quad scan routine on system: 2022/04/18

First 6 rounds: 54 saved data points with old script



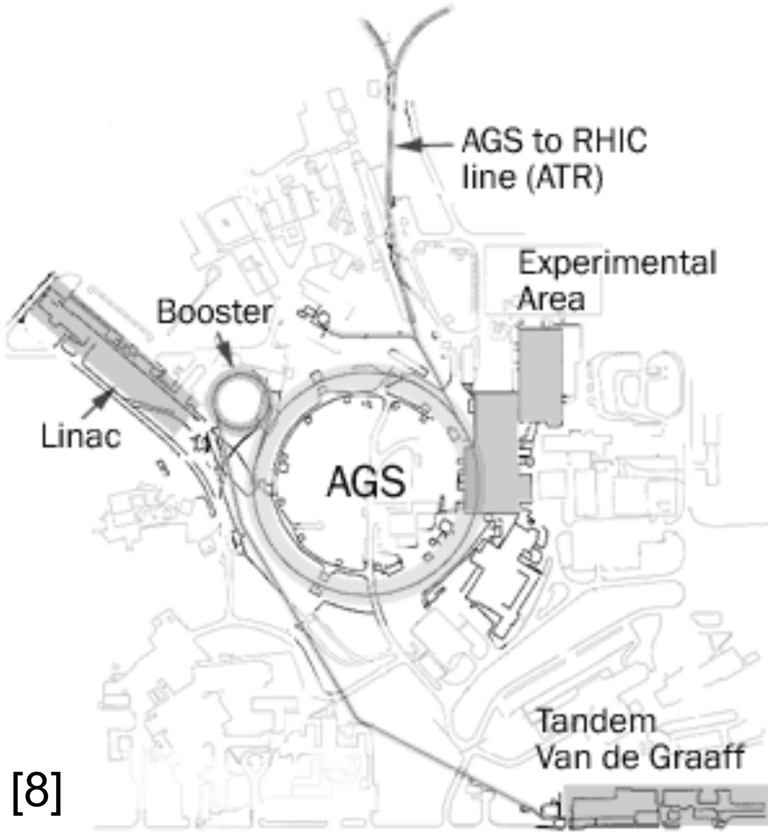
[7]

Remaining 7 rounds: 7 data points using Q3-Q4 settings predicted by NN model



- NN with one hidden layer , ReLU and Tanh activation functions
- Trained NN accuracy on 54 data points: 93.65%
- Trouble getting the small Y RMS region features, maybe Y range is too large
- Tested 7 proposed Q3-Q4 combo settings
- Obtained Y RMS values around 0.3 – 0.4 mm range: satisfactory preliminary results
- Successfully cut scan time by 50%

Brightness control at the Alternating Gradient Synchrotron (AGS)



[8]

- Alternating gradient / strong focusing principle: achieve strong vertical and horizontal focusing of charged particle beam at the same time
- Accelerates proton to 33 GeV in 1960
- 12 super-periods (A to L), 240 main magnets
- Now serves as injector for Relativistic Heavy Ion Collider (RHIC)

Motivation: support for EIC Cooler

- Electron cooling for the EIC requires small incoming emittances
- Necessary pre-cooler at RHIC injection energy (AGS extraction energy)
- Current AGS lacks systematic tuning routine, mostly hand tuned by operators
- Algorithm to better control beam in AGS will be helpful for future EIC cooler
- CeC experiment continues in February 2023

Orbit Response Matrix (ORM)

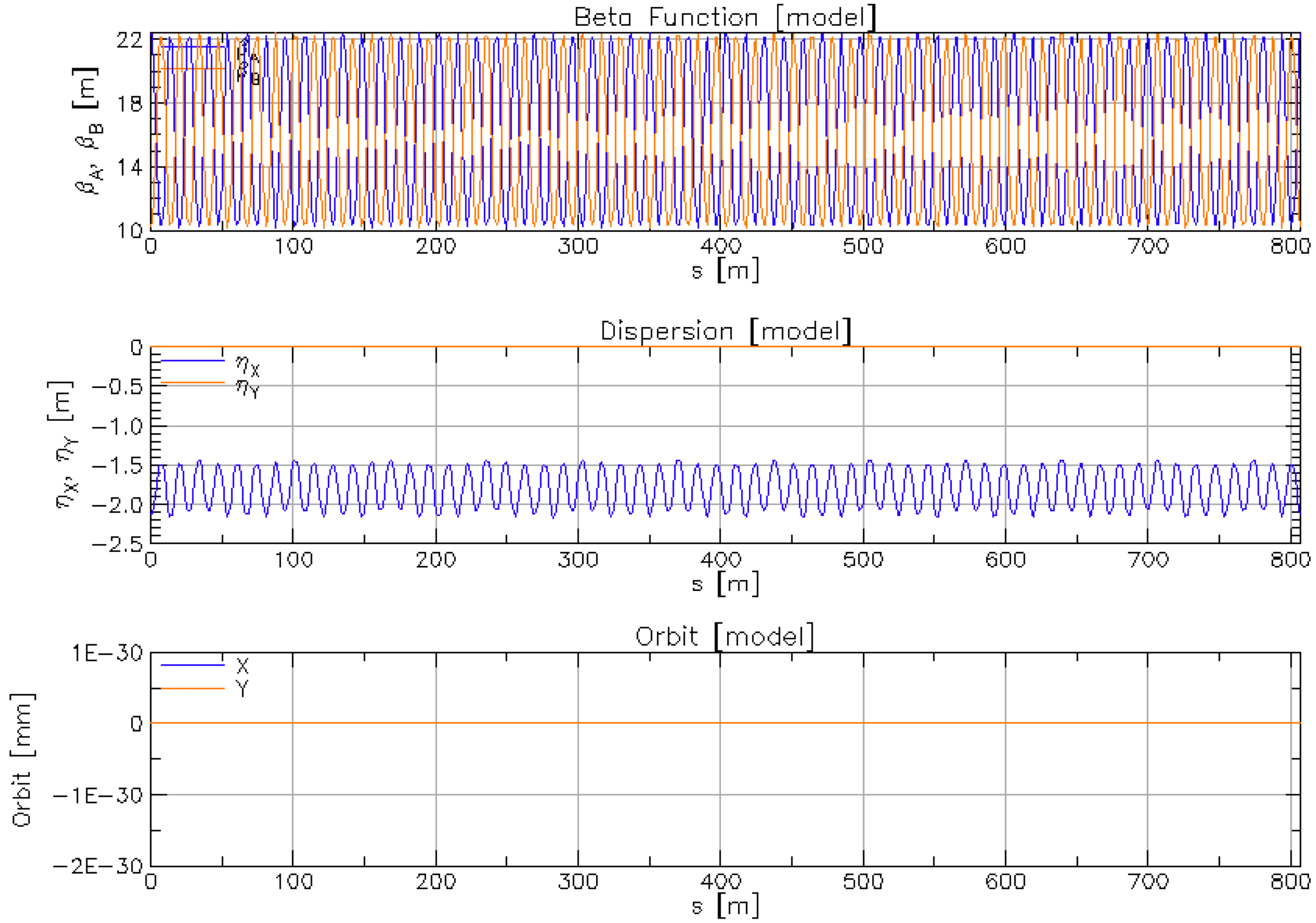
- Mapping $\vec{R}$ between closed orbit measurements and corrector settings
- 72 pick-up electrodes (PUE), 48 horizontal and vertical corrector pairs
- Linear orbit response to corrector change: calculate R matrix by changing each corrector pair separately
- Corrector current $I \rightarrow$ angle θ by calibration factor
- Traditional orbit correction: $\Delta\vec{\theta} = \vec{R}^{-1}\,\Delta\vec{y}$

$$\begin{pmatrix} \Delta\vec{x} \\ \Delta\vec{y} \end{pmatrix} = \vec{R} \begin{pmatrix} \Delta\vec{\theta}_x \\ \Delta\vec{\theta}_y \end{pmatrix}$$

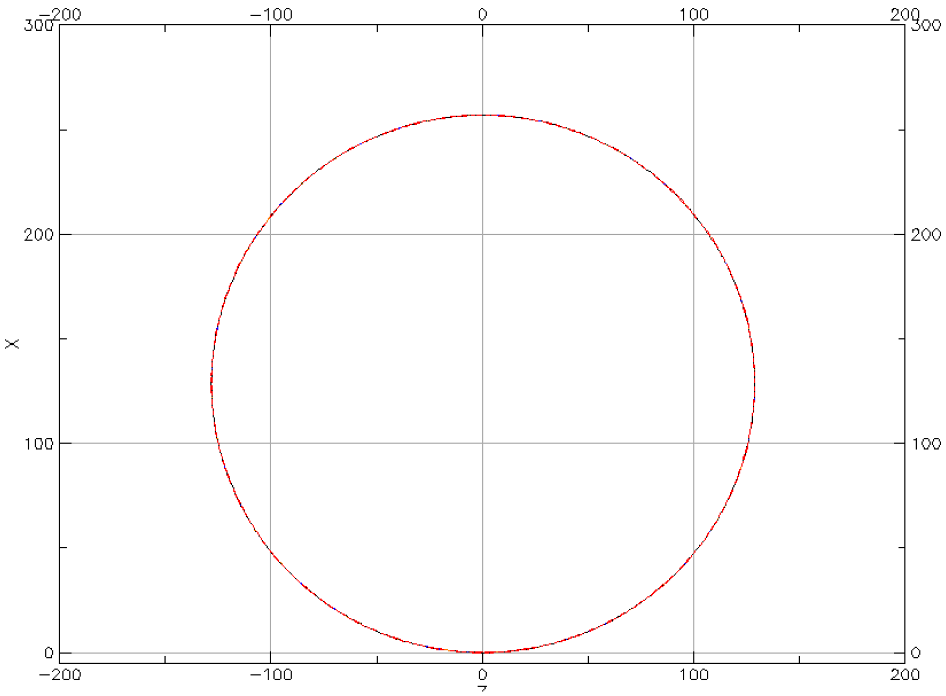
$$\frac{\Delta x_i}{\Delta \theta_j} = R_{ij}$$

MAD-X to BMAD translation

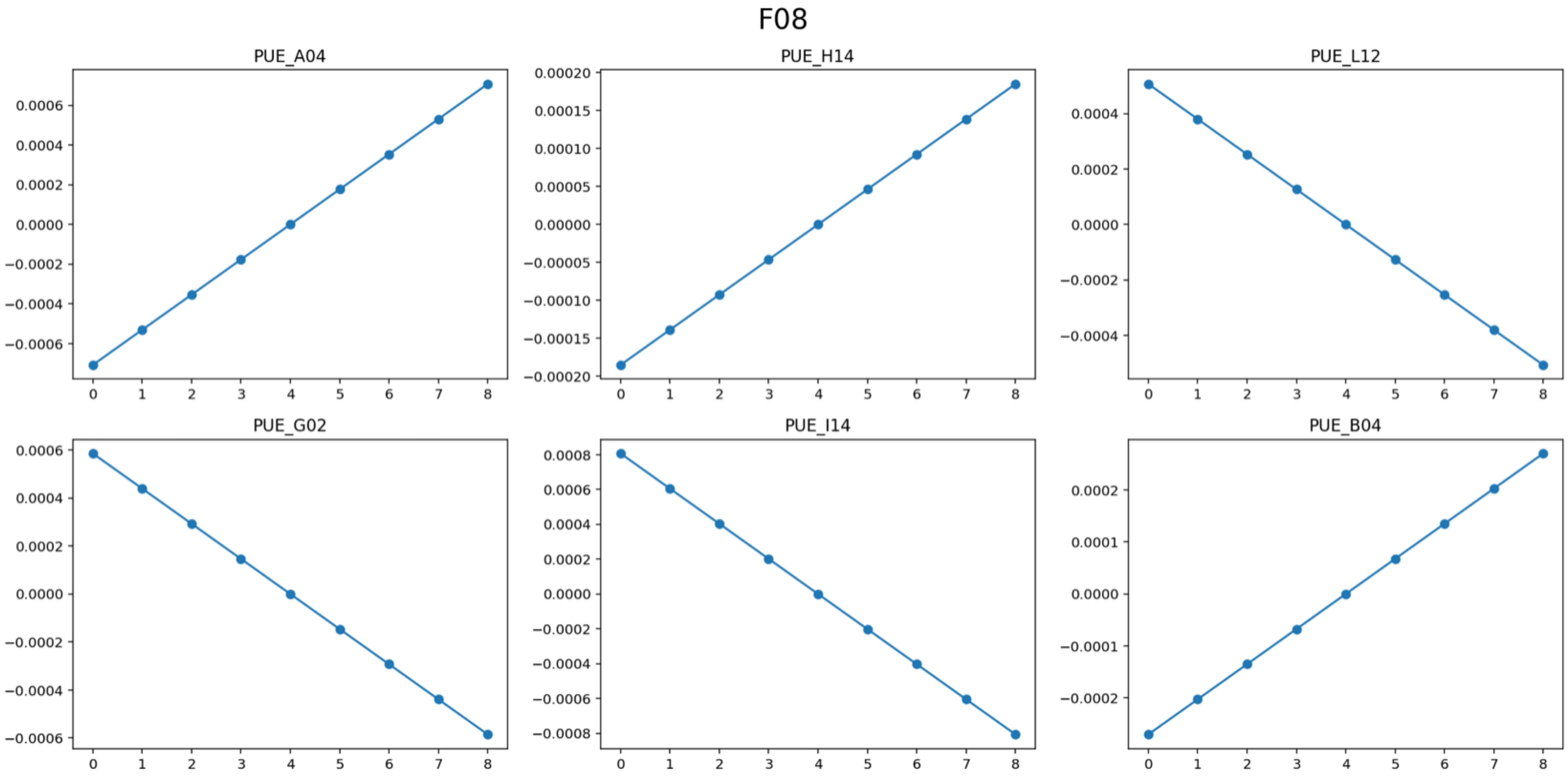
- Successfully translated bare machine to BMAD: ramping in progress
- Can use Python interface (pytao) to run simulations much easier



Floor plan



Orbit Response vs. One Corrector (Sim.)



Use ORM to identify machine errors

- Actual machine with errors (e.g. quadrupole gradient errors, corrector calibration errors, etc.) produce different $\vec{R}_{measured}$ from model/reference machine $\vec{R}_{model}$

$$\Delta R_{ij} = R_{ij}^{model} - R_{ij}^{measured}$$

- Considering all possible sources of errors as a vector $\vec{\nu}$, build response error model $\vec{J}_{model}$

$$\begin{pmatrix} \Delta R_{11} \\ \Delta R_{12} \\ \cdots \\ \Delta R_{n(m-1)} \\ \Delta R_{nm} \end{pmatrix} = J_{model} \begin{pmatrix} \Delta \nu_1 \\ \Delta \nu_2 \\ \cdots \\ \Delta \nu_{N-1} \\ \Delta \nu_N \end{pmatrix}$$

- Reconstruct any $\vec{\nu}$ given known $\Delta\vec{R}$ and $\vec{J}_{model}$

Reconstruct errors using SVD

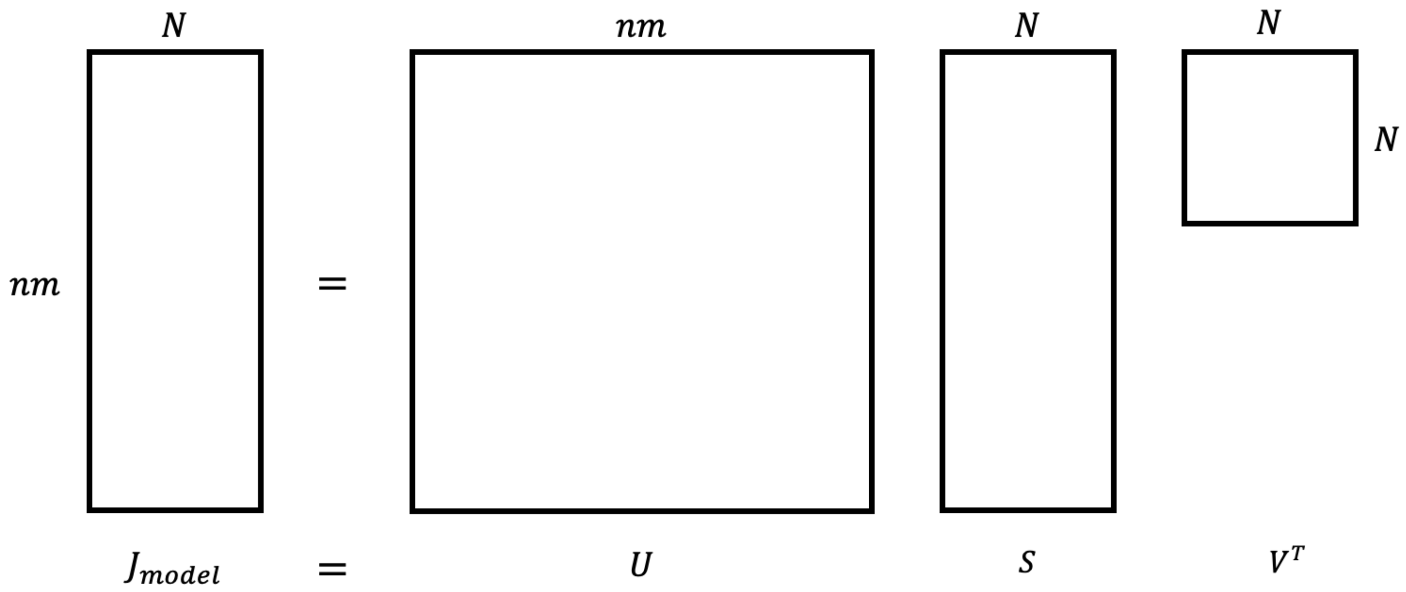
- Traditional tuning routine: perform singular value decomposition (SVD) directly on $\vec{R}$
- Machine error detection: perform SVD on $\vec{J}_{model}$
- Solve for $\Delta\vec{v}$ using $\Delta\vec{R} = \vec{J}_{model}\,\Delta\vec{v}$, where $\vec{J}_{model}$ is not a square matrix

$$J_{model} = USV^T$$

$n = N_{corr}, m = N_{BPM}$

$\Delta\vec{R}: (48 \times 72, 1)$

$\vec{J}_{model}: (3456, N_{error})$



Test case: quadrupole strength error

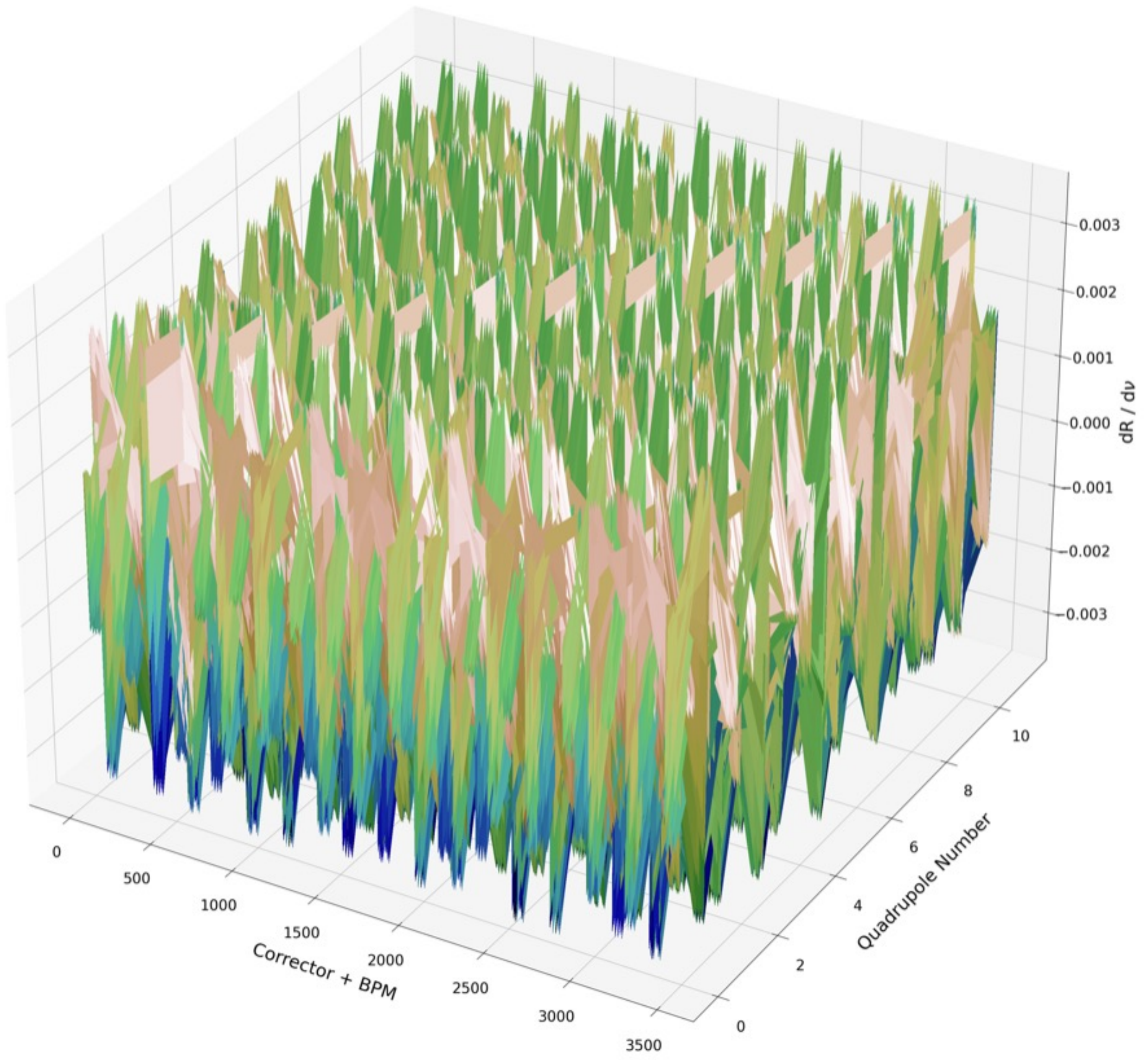
- 24 quadrupoles (12 horizontal, 12 vertical), 1 in each super-period
- Linear orbit response to quadrupole kick change: calculate $\Delta\vec{R} = \vec{R}_{measured} - \vec{R}_{ref}$ by changing each quadrupole separately $\rightarrow J_{ijk} = \frac{\Delta R_{ij}}{\Delta v_k}$
- Quad kick defined with one variable KQH/KQV in MAD-X → variables in BMAD allow separate change of quad kicks

```
tao.cmd('show var quads.x')
```

```
['  Variable                  Slave Parameters          Meas        Model        Design   Useit_opt',
 '  quads.x[1]                QH_F17[K1]            0.0000E+00  -6.5349E-05  -6.5349E-05        T',
 '  quads.x[2]                QH_G17[K1]            0.0000E+00  -6.5349E-05  -6.5349E-05        T',
 '  quads.x[3]                QH_H17[K1]            0.0000E+00  -6.5349E-05  -6.5349E-05        T',
 '  quads.x[4]                QH_I17[K1]            0.0000E+00  -6.5349E-05  -6.5349E-05        T',
 '  quads.x[5]                QH_J17[K1]            0.0000E+00  -6.5349E-05  -6.5349E-05        T',
 '  quads.x[6]                QH_K17[K1]            0.0000E+00  -6.5349E-05  -6.5349E-05        T',
 '  quads.x[7]                QH_L17[K1]            0.0000E+00  -6.5349E-05  -6.5349E-05        T',
 '  quads.x[8]                QH_A17[K1]            0.0000E+00  -6.5349E-05  -6.5349E-05        T',
 '  quads.x[9]                QH_B17[K1]            0.0000E+00  -6.5349E-05  -6.5349E-05        T',
 '  quads.x[10]               QH_C17[K1]            0.0000E+00  -6.5349E-05  -6.5349E-05        T',
 '  quads.x[11]               QH_D17[K1]            0.0000E+00  -6.5349E-05  -6.5349E-05        T',
 '  quads.x[12]               QH_E17[K1]            0.0000E+00  -6.5349E-05  -6.5349E-05        T',
 '  Variable                  Slave Parameters          Meas        Model        Design   Useit_opt']
```

Test case $\vec{J}_{model}$ matrix (horizontal)

- Calculated using $\Delta\upsilon = 40$ for each quadrupole
- Agreement with MAD-X model (redefined every quad individually) was obtained



Reconstruct errors using SVD

- $\vec{U}$ and $\vec{V}$ are square orthogonal matrices: $UU^T = VV^T = I$
- $\vec{S}$ is an $nm \times N$ matrix whose first N diagonal elements are singular values σ of $\vec{J}_{model}$

$$S = \begin{pmatrix} S_N \\ 0 \end{pmatrix} \in \mathbb{R}^{nm \times N}, \quad S_N := diag(\sigma_1, \ldots, \sigma_N, 0, \ldots, 0) \in \mathbb{R}^{N \times N}$$

- $\vec{S}^+$ is pseudoinverse of $\vec{S}$ whose first N diagonal elements are $\frac{1}{\sigma}$

$$S^+ = \begin{pmatrix} S_N^+ \\ 0 \end{pmatrix} \in \mathbb{R}^{N \times nm}, \quad S_N^+ := diag(\frac{1}{\sigma_1}, \ldots, \frac{1}{\sigma_N}, 0, \ldots, 0) \in \mathbb{R}^{N \times N}$$

$$\begin{pmatrix} \Delta\nu_1 \\ \Delta\nu_2 \\ \cdots \\ \Delta\nu_{N-1} \\ \Delta\nu_N \end{pmatrix} = J^+_{model} \begin{pmatrix} \Delta R_{11} \\ \Delta R_{12} \\ \cdots \\ \Delta R_{n(m-1)} \\ \Delta R_{nm} \end{pmatrix} = V S^+ U^T \begin{pmatrix} \Delta R_{11} \\ \Delta R_{12} \\ \cdots \\ \Delta R_{n(m-1)} \\ \Delta R_{nm} \end{pmatrix}$$

Test case: reconstruct errors with $\vec{J}_{model}$

- Case 1: change one quadrupole

```
# reconstruct quad A17 50 - 10
np.dot(V, np.dot(S_inv, np.dot(UT, dR[0])))
```

```
array([ 4.00000000e+01,  5.55111512e-14, -5.32907052e-15,  6.66133815e-14,
       -1.73194792e-14,  6.39488462e-14, -3.59712260e-14,  6.21724894e-14,
       -2.26485497e-14,  6.21724894e-14, -8.88178420e-15,  5.86197757e-14])
```

- Case 2: change two quadrupoles

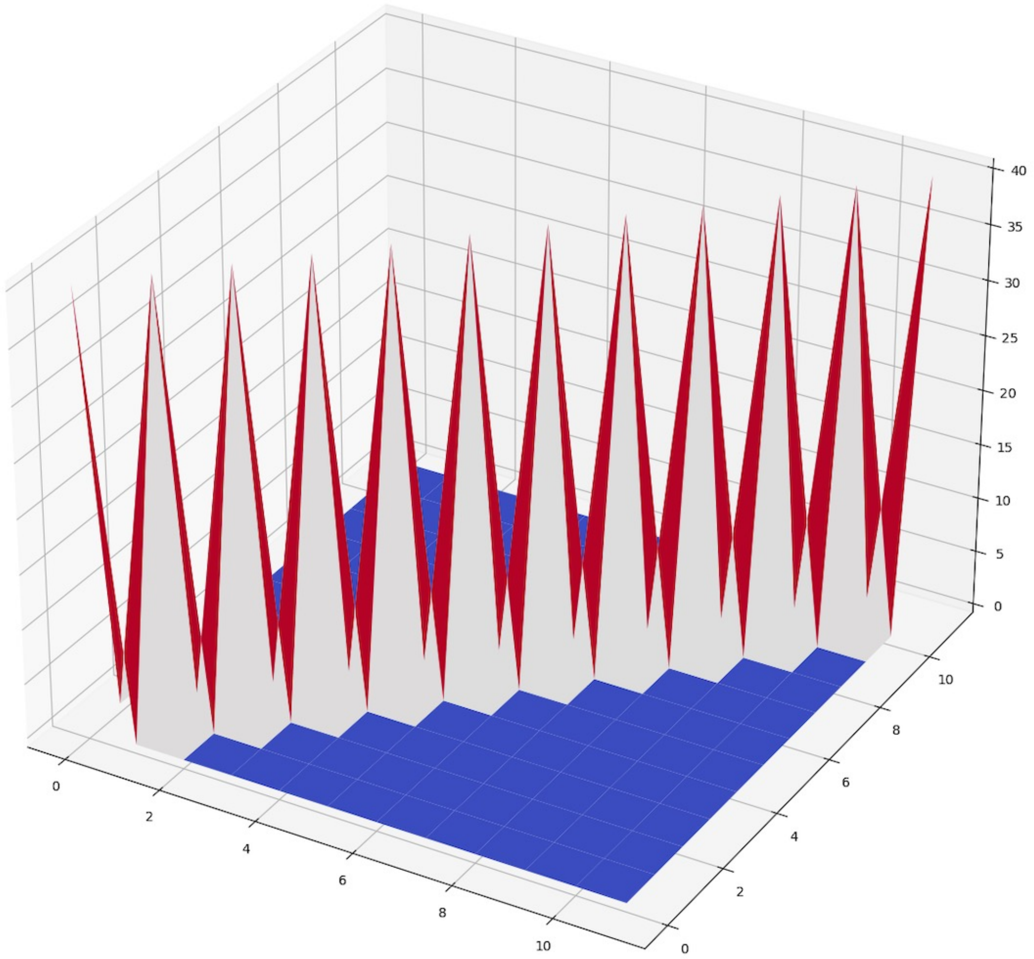
```
# reconstruct AH 50
np.dot(V, np.dot(S_inv, np.dot(UT, dr50)))
```

```
array([ 5.01033941e+01, -2.49280309e-02, -1.11754624e-02, -1.30517756e-02,
       -1.32712155e-02, -1.14236717e-02, -2.45568371e-02,  5.01034603e+01,
        2.23274426e-02,  1.77476325e-02,  1.78368519e-02,  2.26005639e-02])
```

- Case 3: change three quadrupoles

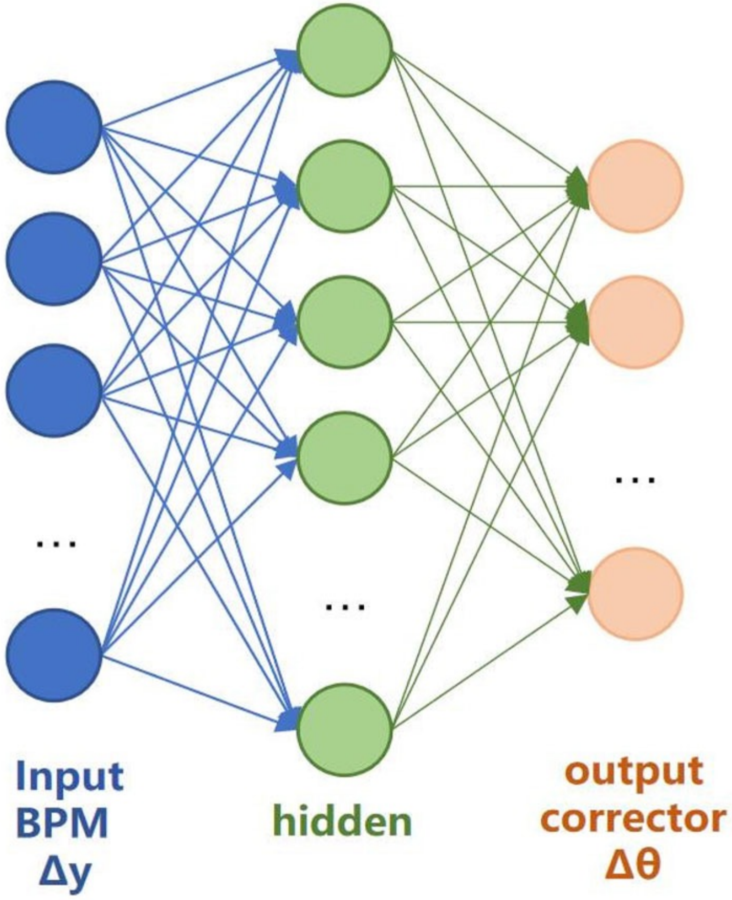
```
# reconstruct B 33 F 17 J 48
np.dot(V, np.dot(S_inv, np.dot(UT, dr50)))
```

```
array([ 1.43131133e-02,  3.25994055e+01, -9.33613404e-03, -7.09123076e-02,
        5.18614771e-03,  1.67488152e+01,  1.08920689e-02, -8.71645682e-02,
       -8.19728759e-03,  4.74841054e+01,  9.22267860e-03, -1.35799339e-01])
```



Satisfactory reconstruction results

Neural Network for real-time ORM

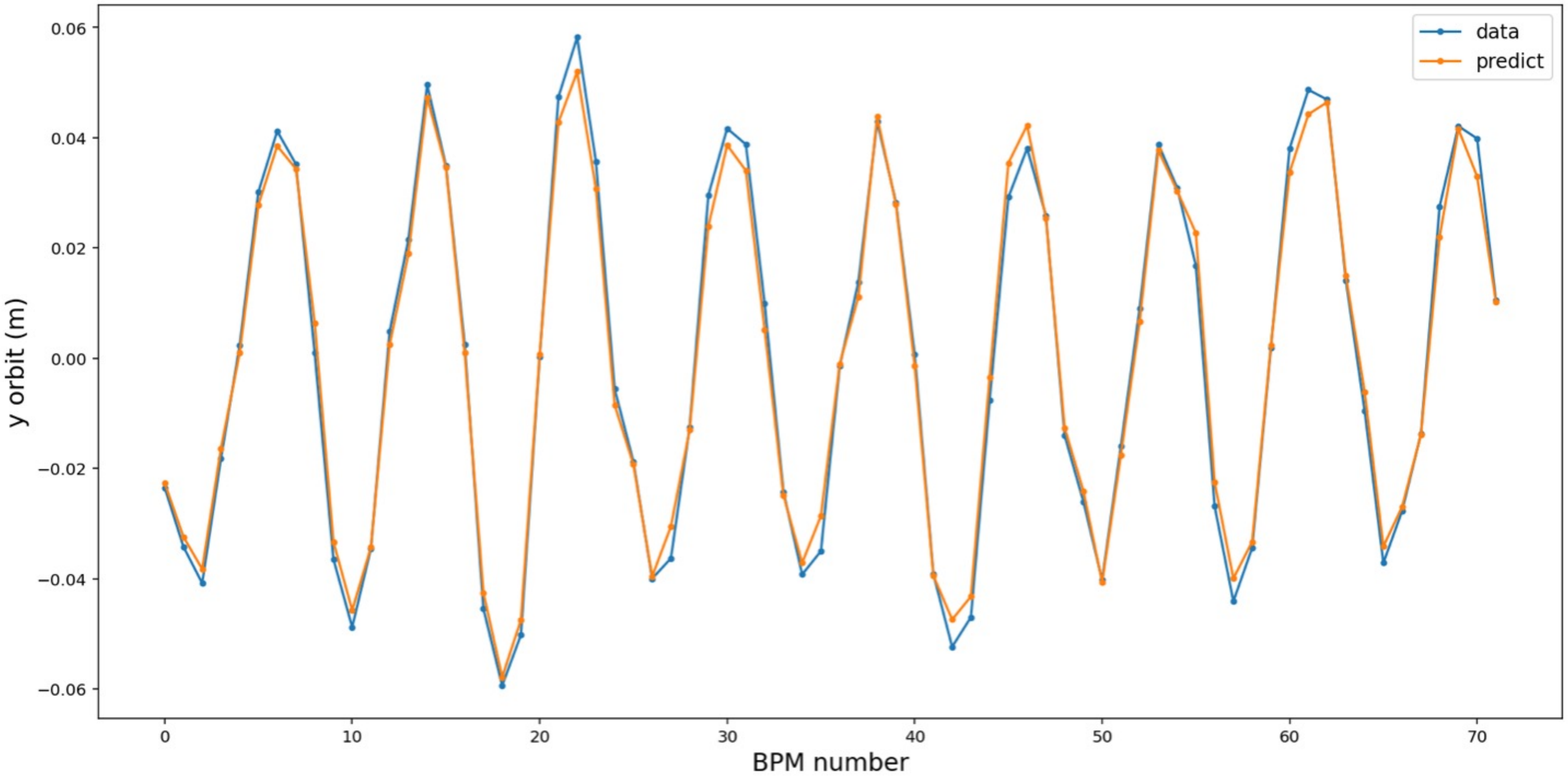


[9]

- Need dedicated machine time to measure ORM $\vec{R}_{measured}$: at least 30 min
- Pre-measured $\vec{R}_{measured}$ gets less accurate with time → orbit drift / brightness drop
- Update ORM with real-time data: build neural network model for $\vec{R}_{measured}$ or $\vec{R}^{-1}_{measured}$
- Can be used to calculate $\Delta\vec{R}$ for machine error reconstruction

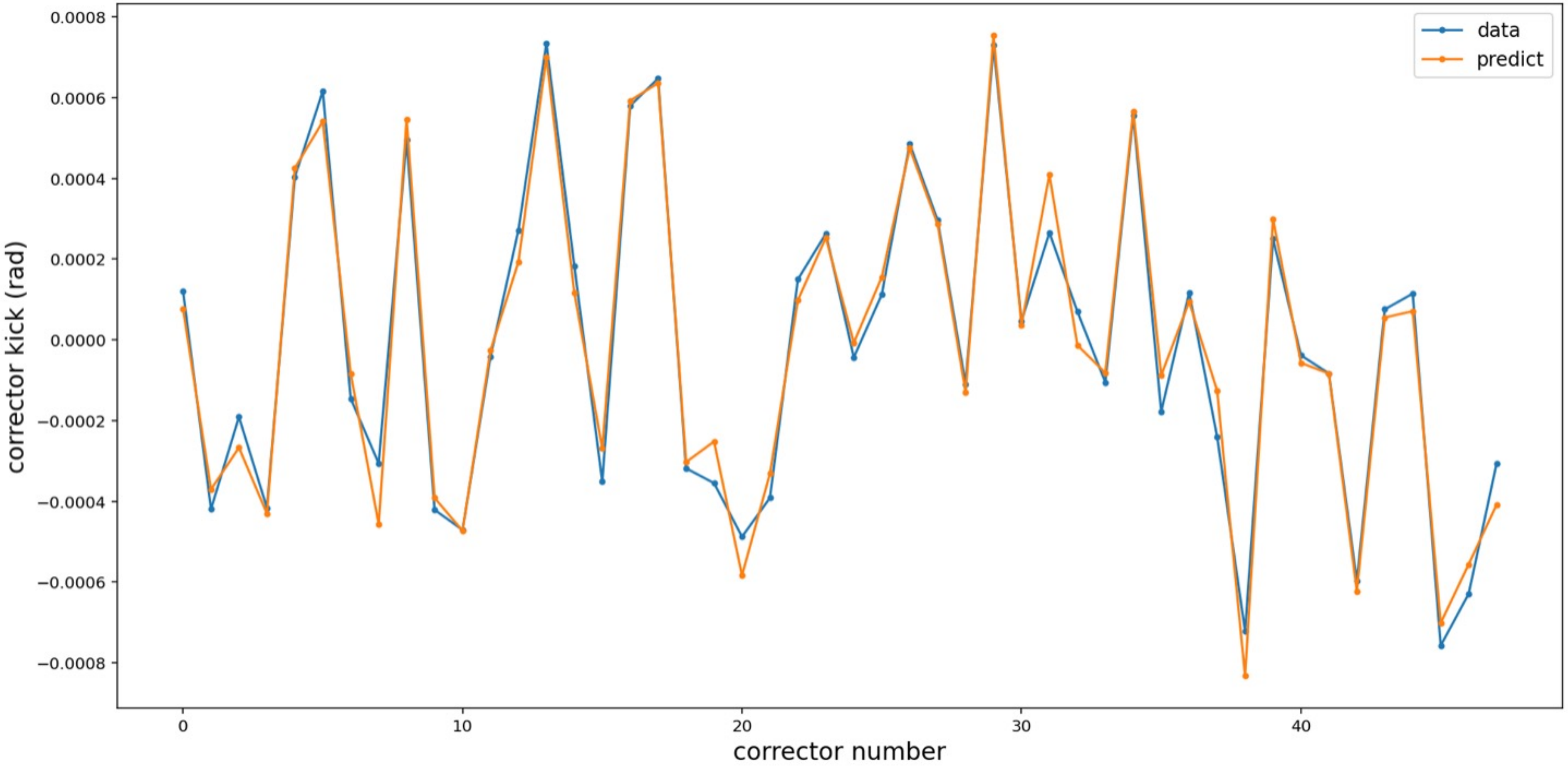
ORM NN model: training results

Input 48 vertical corrector kick → Output 72 y orbit measured at BPM



Inverse ORM NN model: training results

Input 72 y orbit measured at BPM → Output 42 vertical corrector kick



Conclusion

- Neural network can be trained as surrogate models for accelerator beam lines, possible to build digital twin for larger accelerator systems
- Conventional operational routines can be more efficient with help from machine learning
- It shows the significant benefit of incorporating machine learning algorithms into control systems at accelerator facilities

References

- [1] V. Litvinenko et al., “Coherent electron Cooling (CeC) as an EIC cooler”, EIC Collaboration Workshop: Promoting Collaboration on the Electron-Ion Collider, Oct. 9 2020. https://indico.cern.ch/event/949203/contributions/3989899/attachments/2119592/3566940/CeC_for_EIC.pdf
- [2] V. Litvinenko et al., “Coherent electron Cooling (CeC) experiment at RHIC”, CAD MAC, December 1, 2020. https://www.rhichome.bnl.gov/RHIC/Runs/RhicRun20AuAuCeC.pdf
- [3] K. Shih, “Slice Emittance Measurement on CeC Diagnostic Beamline”, Jan. 21 2022.
- [4] J. Edelen, “Inverse Models for Implicit Tuning of the ATR Line and Machine Learning @ RadiaSoft”, BNL Machine Learning Team meeting, Mar. 25 2022.
- [5] A shallow neural network for simple nonlinear classification, https://scipython.com/blog/a-shallow-neural-network-for-simple-nonlinear-classification/, Accessed on May 14 2022.
- [6] S. Sharma, Activation Functions in Neural Networks, https://towardsdatascience.com/activation-functions-neural-networks-1cbd9f8d91d6, Accessed on May 14 2022.
- [7] W. Lin, Y. Jing, et al., “Simulation Studies and Machine Learning Applications at the Coherent electron Cooling experiment at RHIC”, in *Proc. IPAC'22*, Bangkok, Thailand, Jun. 2022, pp. 2387-2390.
- [8] Alternating Gradient Synchrotron, https://www.bnl.gov/rhic/ags.php, Accessed on Sep. 6 2022.
- [9] Y. Bai et al., “Research on the slow orbit feedback of BEPCII using machine learning”, Rad. Det. Tech. Meth. 6, 179-186 (2022).