A HYBRID PHYSICS AND DATA-DRIVEN MODELING FRAMEWORK FOR ACCELERATORS WITH AUTOMATIC DIFFERENTIATION*

Y. Hao†, H. Alamprese, W. Fung, C. Ratcliff, Facility for Rare Isotope Beams, East Lansing, MI, USA
J. Wan, Institute of High Energy Physics, Beijing, China
J. Qiang, Lawrence Berkeley National Laboratory, Berkeley, CA, USA

Abstract

In modern accelerator modeling, many lattice components can be accurately described using established first-principles physics. However, certain intricate effects, such as complex boundary conditions, collective interactions, and self-fields, remain difficult to model reliably and efficiently from theory alone. At the same time, high-resolution beam position and profile measurements provide rich information about these poorly understood dynamics. In this paper, we present a hybrid modeling framework with built-in automatic differentiation, designed to seamlessly integrate physics-based lattice models with data-driven representations of complex effects. This approach improves predictive accuracy, enables gradient-based optimization, and offers a practical path toward more faithful digital twins of accelerator systems.

INTRODUCTION

Accurate modeling of accelerator beam dynamics is essential for optimizing machine performance and enabling reliable operation of modern accelerator facilities. Traditionally, accelerator models are constructed from first-principles beam physics and detailed lattice descriptions, followed by parameter fitting using diagnostic measurements. While this approach has been highly successful, some important effects remain difficult to model accurately due either to incomplete physical understanding or to prohibitive computational cost. Examples include collective effects, complex electromagnetic structures such as undulators, RF cavities, and magnets with sophisticated geometries.

Recently, machine learning (ML) based surrogate models have emerged as a promising approach for improving accelerator modeling. In this framework, difficult-to-model or computationally expensive physical processes are replaced by neural-network (NN) or Gaussian Process trained using either simulation data or experimental measurements.

One recent example is the ML representation of the space-charge Hamiltonian for beam tracking [1], using a kick-drift model, with macroparticles drifting through short beamline segments, followed by space-charge kicks generated from a Hamiltonian map predicted by a NN using the instantaneous beam distribution as input. The Hamiltonian formulation preserves the symplectic structure of the tracking algorithm even when the neural-network approximation does not perfectly reproduce the true self-field Hamiltonian.

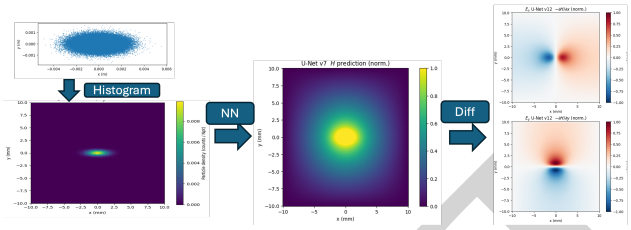

Figure 1: The flow chart of the ML workflow for space charge calculation.

However, the ML-based space-charge models does not naturally work with auto-differentiation package to get the gradients of simulation outputs with respect to the initial beam coordinates or lattice parameters, such as drift lengths or quadrupole strengths. This limitation prevents the direct application of gradient-based optimization and sensitivity analysis in hybrid physics-ML simulations.

In this work, we present an automatic-differentiation-enable implementation of a hybrid accelerator tracking model that combines conventional beam optics with a NN-based space-charge solver. The proposed approach preserves differentiability through both the physics-based tracking and the ML space-charge calculations, enabling gradient evaluation for the beam dynamics studies.

ML SPACE CHARGE MODEL

Following our earlier work [1], we retrained the NN space-charge model using identical computational domains for both the spectral Poisson solver and the grid boundary for the NN model (20 mm×20 mm). The workflow of the ML-based space-charge solver is illustrated in Figure 1. The transverse macroparticle distribution is first converted into a density histogram, which serves as the input to the NN. The NN predicts the space-charge Hamiltonian map, from which the space-charge kicks are obtained by spatial differentiation.

To evaluate the accuracy of surrogate model, we consider a bi-Gaussian transverse beam distribution with $\sigma_x = 3\sigma_y$. Figure 2 compares the electric field predicted by the neural network with the analytical Bassetti–Erskine solution. The surrogate model reproduces the field with better than 10% accuracy while preserving the symplectic structure of the particle tracking.

AD-ENABLED HYBRID MODEL

To demonstrate the automatic-differentiation capability of the NN-based space-charge solver and its integration with

* Work supported by the DOE Office of Science, Office of High Energy Physics, with Award No. DE- SC0024170 and No. DE-SC0023722. The work of J. Q. is also supported by the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.
† haoyue@msu.edu

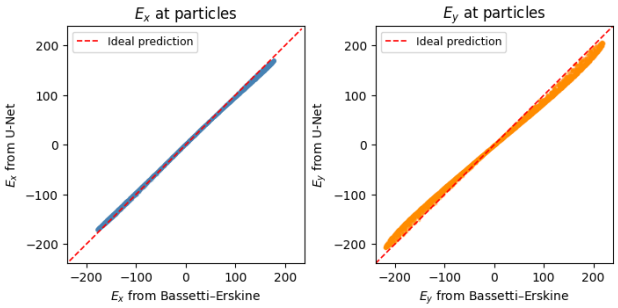

Figure 2: Field accuracy at locations of macro-particles from space charge AI model.

physics-based beam dynamics, we consider the same FODO lattice studied in Ref [1]. The lattice transports a 1 GeV proton beam with a beam current of 200 A. The FODO cell consists of two quadrupoles with length 0.1 m and strengths $K = \pm 29.6\ \mathrm{m}^{-2}$, separated by a drift space of length 0.4 m.

The space-charge effect is modeled using kick-drift splitting. Space-charge kicks are inserted every 0.05 m in drift sections and every 0.025 m inside quadrupoles. The beam distribution is matched at the center of the drift space between the defocusing and focusing quadrupoles with normalized emittances of 1 µm in both transverse planes.

To enable automatic differentiation through the hybrid tracking model, the PyTorch implementation of the neural network was converted into a Julia `NNlib` implementation compatible with the Enzyme automatic differentiation framework within the JuTrack code [2]. The neural network takes histogram representation of the transverse beam distribution an input. Conventional hard-bin histograms use discontinuous step-function shape functions and are therefore incompatible with automatic differentiation. To maintain differentiability, we replace the hard-bin interpolation with smooth bicubic Catmull–Rom interpolation [3], ensuring that the space-charge kicks obtained from the Hamiltonian remain differentiable with respect to particle coordinates and lattice parameters.

Jacobian of the Tracking Map

One important application of automatic differentiation in space-charge simulations is the evaluation of the linearized one-turn map and the corresponding tune shift. To compute the Jacobian matrix, we introduce a test particle that experiences the self-consistent space-charge field generated by the beam while neglecting its own contribution to the collective field. The test particle is initialized at $\mathbf{x}_0 = (x, p_x, y, p_y) = (0, 0, 0, 0)$. Traditionally, the Jacobian matrix can be obtained using finite-difference (FD) methods by perturbing each phase-space coordinate independently. Using central finite differences, each column of the Jacobian requires two tracking evaluations, resulting in a total of eight evaluations for the four-dimensional transverse phase space. In principle, this approach provides second-order accuracy with respect to the perturbation step size. However, known challenge of finite-differences methods remains, i.e. the sensitive to the perturbation step size while the optimum setting is not known beforehand. The Jacobian matrix obtained from finite differences using perturbation step 1×10^{-6} is shown in Table 1. Only a limited number of significant digits are displayed due to space constraints.

Table 1: Jacobian Matrix from FD

-1.047	0.938	-3.8×10^{-4}	2.5×10^{-3}
-3.362	2.058	1.2×10^{-2}	1.0×10^{-2}
-6.1×10^{-3}	4.5×10^{-4}	2.122	0.941
4.7×10^{-3}	4.4×10^{-3}	-3.643	-1.143

Table 2: Jacobian Matrix from Enzyme

-1.043	0.938	-2.0×10^{-4}	2.5×10^{-3}
-3.353	2.058	1.2×10^{-2}	1.0×10^{-2}
-6.1×10^{-3}	4.5×10^{-4}	2.127	0.941
4.7×10^{-3}	4.4×10^{-3}	-3.649	-1.144

Alternatively, we can use the `Enzyme.jl` automatic-differentiation (AD) package [4] to evaluate the Jacobian matrix directly from the computational graph of the tracking routine. AD computes the local derivative analytically through the chain rule applied to every operation in the tracking algorithm, including the interpolation of the frozen space-charge Hamiltonian map. As a result, the AD Jacobian is not sensitive to the perturbation step size. The Jacobian matrix obtained using Enzyme forward-mode AD is shown in Table 2.

We observed that the Jacobians from FD and AD differ at an approximately 1×10^{-3}. This discrepancy is primarily attributed to the numerical representation of the neural-network-generated Hamiltonian maps and the interpolation procedure used for the space-charge kicks. In the present implementation, the Hamiltonian tables produced by the neural network are stored in `Float32` precision, while the kicks are evaluated using Catmull–Rom bicubic interpolation. Reducing the finite-difference step size below approximately 10^{-6} does not significantly improve the agreement between the FD and AD Jacobians, and the discrepancy saturates near the 10^{-3} level. This behavior indicates that the dominant error source is not finite-difference truncation error, but rather the intrinsic piecewise-polynomial structure and finite numerical precision of the interpolated neural-network field representation.

The automatic-differentiation framework can be further applied to evaluate the dependence of the lattice tune on beam current through the tune-extraction function constructed from the one-turn Jacobian matrix. Figure 3 shows the horizontal tune calculated at different beam currents (blue points), together with the tune slopes obtained directly from AD (short red lines). The AD-derived slopes exhibit good agreement with the overall trend of the current-tune dependence, demonstrating that the differentiable tracking framework can correctly capture the sensitivity to beam intensity.

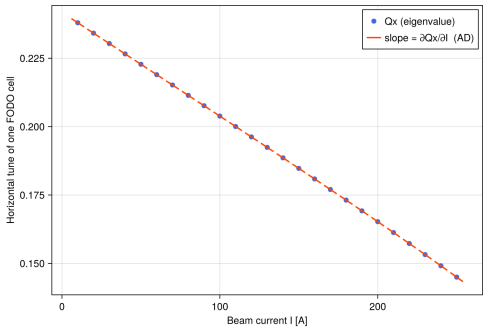

Figure 3: The linear horizontal tune as function of the current

Auto Differentiation wrt. Lattice Parameters

One of the more challenging applications of automatic differentiation is the evaluation of beam-emittance sensitivity with respect to lattice parameters in the presence of self-consistent space-charge effects. In this study, we consider the differentiated lattice-parameter vector

$$\theta = \left[L_{D1},\, L_{Q1},\, K_{Q1},\, L_{D2},\, L_{Q2},\, K_{Q2},\, L_{D3}\right], \qquad (1)$$

where drift lengths, quadrupole lengths are illustrated in Fig. 4.

Unlike the frozen-field test-particle Jacobian calculation, variations of the lattice parameters modify the beam distribution itself during transport. Consequently, the beam distribution arriving at each space-charge kick location changes, which in turn modifies the Hamiltonian generated by the neural-network space-charge model. Therefore, the Hamiltonian values on the computational grid become implicit functions of the differentiated lattice parameters. The total derivative of an observable f with respect to a lattice parameter θ_i must therefore contain two contributions,

$$\frac{df}{d\theta_i} = \left.\frac{\partial f}{\partial \theta_i}\right|_{H_k \text{ fixed}} + \sum_k \frac{\partial f}{\partial H_k}\frac{\partial H_k}{\partial \theta_i}, \qquad (2)$$

where H_k denotes the space-charge Hamiltonian predicted by the neural network at the kth kick location. The first term corresponds to the direct optics contribution with the NN-generated Hamiltonian frozen, similar to the test-particle Jacobian calculation discussed previously. The second term represents the self-consistent field response arising from changes in the beam distribution and therefore changes in the NN-generated Hamiltonian.

The NN maps the normalized transverse charge-density distribution to the space-charge Hamiltonian according to $H_{\text{sc}} = \mathcal{N}]!\mathcal{N}(\rho, w)$ where w represents the fixed trained network weights, and ρ is the transverse beam-density distribution. The U-Net implementation employs convolution and transpose-convolution operations provided by the Julia package `NNlib`. Direct differentiation through these operations using Enzyme encountered unsupported low-level BLAS and convolution kernels. To overcome this limitation, the differentiation through the neural network was implemented using a custom forward-mode Jacobian vector product (JVP).

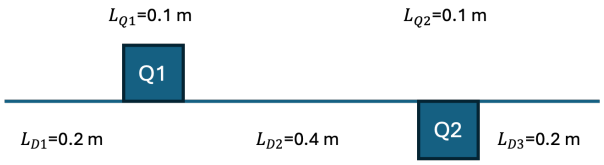

Figure 4: Lattice parameters for differentiation in FODO cell.

Table 3: Comparison between AD and FD gradients for the normalized horizontal emittance growth at 200 A.

θ_i	AD	FD	Rel. Error
L_{D1}	3.04×10^{-2}	$+3.04 \times 10^{-2}$	2.0×10^{-4}
L_{Q1}	-1.11×10^{-1}	-1.12×10^{-1}	4.0×10^{-3}
K_{Q1}	-4.33×10^{-4}	-4.34×10^{-4}	3.4×10^{-3}
L_{D2}	-5.32×10^{-3}	-5.33×10^{-3}	3.6×10^{-4}
L_{Q2}	-8.53×10^{-3}	-8.54×10^{-3}	2.3×10^{-4}
K_{Q2}	1.14×10^{-5}	1.14×10^{-5}	7.8×10^{-4}
L_{D3}	-5.66×10^{-3}	-5.66×10^{-3}	1.8×10^{-4}

For an input tangent $\dot{\rho}$, the custom JVP evaluates to

$$\dot{H}_{\text{sc}} = J_{\mathcal{NN}}(\rho)\,\dot{\rho} = \frac{\partial \mathcal{NN}}{\partial \rho}\dot{\rho}, \qquad (3)$$

where $J_{\mathcal{NN}}$ is the Jacobian of the neural network with respect to its input density distribution. In practice, the JVP is constructed explicitly in a layer-by-layer manner for the entire U-Net architecture. This layer-wise tangent propagation enables Enzyme to correctly propagate derivatives through the full hybrid physics-ML tracking model while preserving the self-consistent coupling between lattice optics and space-charge dynamics.

A comparison between the AD and FD gradients for the normalized horizontal emittance growth is shown in Table 3. Good agreement is observed after properly accounting for the parameter dependence of the NN-generated Hamiltonian during the differentiation process.

SUMMARY

In this paper, we report on the implementation of automatic differentiation in a hybrid physics–NN model for both the Jacobian of beam transport and the sensitivity of beam quality to machine-parameter variations. When the parameter variation does not change the neural-network weights, the AD procedure reduces to the case of a pure physics model, blending in the rest of physics models. However, when the neural-network output itself changes with the differentiated parameters, differentiation through the NN becomes necessary. If the AD package cannot automatically trace through the network, as in the demonstrated case of Enzyme, a customized JVP is required to enable the correct AD process, which demands a layer-by-layer JVP definition of the activation functions. Using this example, we demonstrate that there is no fundamental obstacle to maintaining auto-differentiation capability in a hybrid tracking process.

REFERENCES

[1] J. Wan, J. Qiang, and Y. Hao, "Symplectic machine learning model for fast simulation of space-charge effects", *Phys. Rev. Accel. Beams*, vol. 28, p. 074602, 2025. doi:10.1103/bhpv-bcqk

[2] J. Wan, H. Alamprese, C. Ratcliff, J. Qiang, and Y. Hao, "JuTrack: A Julia package for auto-differentiable accelerator modeling and particle tracking," *Comput. Phys. Commun.*, vol. 309, p. 109497, Apr. 2025. doi:10.1016/j.cpc.2024.109497

[3] Bicubic spline method, https://en.wikipedia.org/wiki/CatmullRom_spline

[4] W. Moses and V. Churavy, "Instead of rewriting foreign code for machine learning, automatically synthesize fast gradients," in *Proc. NeuriPS'20*, Vancouver, Canada, May 2020, pp. 12472–-12485, 2020. doi:10.5555/3495724.3496770