

Symplectic machine learning model for fast simulation of space-charge effects

Jinyu Wan^{1,*}, Ji Qiang², and Yue Hao^{1,†}

¹*Facility for Rare Isotope Beams, Michigan State University, East Lansing, Michigan 48824, USA*

²*Lawrence Berkeley National Laboratory, Berkeley, California 94720, USA*



(Received 28 February 2025; accepted 16 July 2025; published 31 July 2025)

Symplectic simulation of space-charge effects is crucial for the design and operation of high-intensity particle accelerators. Traditional methods for simulating these effects are often computationally expensive, resulting in significant overhead. In this work, we introduce a generative model based on a U-Net architecture within a generative adversarial network framework to efficiently simulate space-charge effects. The model is trained to predict the transverse multiparticle space-charge Hamiltonian, which can be physically computed using a gridless spectral method. The one-step symplectic transverse transfer map for the particles is then obtained by differentiating the predicted Hamiltonian. Benchmarking results demonstrate that this generative model achieves an order of magnitude higher computational efficiency compared to the spectral method, providing a highly efficient alternative for simulating space-charge effects with a large number of particles. By maintaining symplecticity, the model effectively preserves the phase-space structure and mitigates nonphysical errors in long-term simulations. This model has been integrated into JuTrack, a novel autodifferentiable accelerator modeling code developed in the Julia programming language.

DOI: [10.1103/bhvp-bcqq](https://doi.org/10.1103/bhvp-bcqq)

I. INTRODUCTION

The advancement of particle accelerators has significantly expanded the frontiers of physics research [1–3] and enabled various applications in industry [4–6]. As the intensity of particle beams increases, space-charge effects from the Coulomb interactions inside the beam become a critical factor influencing beam dynamics, such as shifts of synchrotron and betatron frequencies, emittance growth, halo formation, and beam instabilities that can degrade the performance of accelerators [7].

Accurate simulation of space-charge effects is important for the design, optimization, and operation of high-intensity accelerators. Because of the collective nature of the space-charge forces, one has to consider the entire distribution of the particles, which is much more complicated than single-particle dynamics. Particle-in-cell (PIC) simulations, which use a computational grid to obtain the charge-density distribution to solve the Poisson equation, are widely used to calculate the space-charge forces [8–10]. However, PIC

codes usually do not satisfy the symplectic condition of multiparticle beam dynamics, leading to nonphysical errors in long-term tracking. To preserve symplecticity, a gridless spectral method was proposed to directly study the multiparticle Hamiltonian and calculate the space-charge forces [11]. This method preserves the phase-space structure and shows much less numerical emittance growth than the PIC model. The computational cost for these methods usually scales worse than the N_p , which can be a significant challenge for simulating a large number of particles.

In recent years, there has been a growing interest within the accelerator community in utilizing machine learning models as alternatives to traditional simulation tools to accelerate the simulation by learning high-fidelity simulation data [12–16]. Specifically, machine learning-enhanced collective effect simulations have become a notable area, such as the surrogate modeling of coherent synchrotron radiation [17–19] and space-charge effects [20]. Some specific machine learning methods are also developed, such as symplectic neural network that maintains the symplectic condition [21] and modular simulation calculations with physics constraints [22,23]. Machine learning algorithms, such as neural networks, have demonstrated remarkable capabilities in learning complex mappings from input data to desired outputs, offering potential reductions in computational effort. Moreover, generative models such as generative adversarial networks (GANs) [24] further enhance these capabilities by learning data distributions and generating high-fidelity samples in deep learning tasks [25,26].

*Contact author: wan@frib.msu.edu

†Contact author: haoy@frib.msu.edu

Published by the American Physical Society under the terms of the [Creative Commons Attribution 4.0 International license](https://creativecommons.org/licenses/by/4.0/). Further distribution of this work must maintain attribution to the author(s) and the published article's title, journal citation, and DOI.

In this work, we propose a generative model based on a U-Net [27] architecture integrated within a GAN framework to efficiently learn the space-charge effects in particle accelerators. The U-Net serves as a deep autoencoder composed of multiple layers of convolutional neural networks [28], enabling the model to capture complex spatial features and mappings between input and output data. By training the model with randomly generated beam distribution data, it effectively learns the hidden correlations between the charge density distribution and the Hamiltonian of space charge from macroparticles, whose true value is calculated with a gridless spectral method proposed in [11]. The one-step symplectic transfer map for the particles is then obtained by differentiating the predicted Hamiltonian, which ensures the preservation of symplecticity in the simulation. This approach not only accelerates the simulation process by reducing computational overhead but also maintains the symplectic properties of the system, providing a faster alternative for long-term beam dynamics studies. We have integrated this machine learning-based space-charge effects model into JuTrack [29], a novel Julia-based particle tracking code specially designed with automatic differentiation capability, which has been integrated into various simulation codes such as Cheetah [20], SciBMad [30], and a symplectic space-charge code [31].

This paper is organized as follows: In Sec. II, we provide the theoretical background on space-charge effects and the gridless spectral method. Section III details the methodology, including the preparation of the training data, the model architecture, and the training procedure. In Sec. IV, we demonstrate the performance of our model. Section V is the conclusion.

II. SYMPLECTIC MODEL OF SPACE-CHARGE FORCES

In this work, our objective is to develop a machine learning model that learns the symplectic transfer map of space-charge forces in particle accelerators. An intuitive approach is to predict the Hamiltonian function of the system. By modeling the Hamiltonian, the self-consistent space-charge simulation is solved by the following equations:

$$\frac{d\mathbf{r}_i}{ds} = \frac{\partial H}{\partial \mathbf{p}_i} \quad (1)$$

$$\frac{d\mathbf{p}_i}{ds} = -\frac{\partial H}{\partial \mathbf{r}_i}, \quad (2)$$

where $H(\mathbf{r}_1, \mathbf{p}_1, \mathbf{r}_2, \mathbf{p}_2, \dots, s)$ is the Hamiltonian of the system, and $\mathbf{r}_i$ and $\mathbf{p}_i$ represent the canonical coordinates and momenta of the i th particle, respectively. The conjugate coordinate-momentum pairs inherently preserve symplecticity in the simulation.

The Hamiltonian of the system can be written as $H = H_1 + H_2$, where $H_1(\mathbf{r}, \mathbf{p})$ includes the external fields from the accelerator beamline elements, and $H_2(\mathbf{r})$ represents the space-charge Hamiltonian. For a coasting beam, H_2 can be written as [31],

$$H_2 = \frac{K}{4} \sum_{i=1}^{N_p} \sum_{j=1}^{N_p} \varphi(\mathbf{r}_i, \mathbf{r}_j), \quad (3)$$

where $K = qI/(2\pi\epsilon_0 p_0 v_0^2 \gamma_0^2)$ is the generalized perveance, q is the charge of a particle, I is the beam current, ϵ_0 is the dielectric constant in vacuum, p_0 , v_0 , and γ_0 are the momentum, velocity, and relativistic factor of the reference particle, respectively, and φ is the space-charge Coulomb interaction potential.

As in [31], we assume that the coasting beam is inside a perfectly conductive rectangular pipe and consider only the transverse plane. The electric Coulomb potential can be obtained by solving the Poisson equation,

$$\frac{\partial^2 \phi}{\partial x^2} + \frac{\partial^2 \phi}{\partial y^2} = -4\pi\rho, \quad (4)$$

where ϕ is the electric potential and ρ is the charge density distribution of the beam.

Following the boundary conditions of the perfectly conducting rectangular pipe, using a Galerkin spectral approximation method [32], the space-charge Hamiltonian H_2 is obtained as [33],

$$H_2 = 4\pi \frac{K}{ab} \frac{1}{N_p} \sum_{i=1}^{N_p} \sum_{j=1}^{N_p} \sum_{l=1}^{N_l} \sum_{m=1}^{N_m} \frac{1}{\gamma_{lm}^2} \sin(\alpha_l x_j) \sin(\beta_m y_j) \sin(\alpha_l x_i) \sin(\beta_m y_i) \quad (5)$$

where N_l and N_m are the number of modes in the horizontal and vertical plane, respectively. $\alpha_l = l\pi/a$, $\beta_m = m\pi/b$, a is the horizontal width of the pipe, b is the vertical width of the pipe, and $\gamma_{lm}^2 = \alpha_l^2 + \beta_m^2$.

The one-step symplectic transfer map of the i th particle is then obtained,

$$p_{xi}(\tau) = p_{xi}(0) - \tau \frac{K}{2} \sum_{l=1}^{N_l} \sum_{m=1}^{N_m} \phi^{lm} \alpha_l \cos(\alpha_l x_i) \sin(\beta_m y_i) \quad (6)$$

$$p_{yi}(\tau) = p_{yi}(0) - \tau \frac{K}{2} \sum_{l=1}^{N_l} \sum_{m=1}^{N_m} \phi^{lm} \beta_m \sin(\alpha_l x_i) \cos(\beta_m y_i), \quad (7)$$

where ϕ^{lm} is the space-charge potential in the spectral domain,

$$\phi^{lm} = 4\pi \frac{4}{ab} \frac{1}{N_p} \sum_{j=1}^{N_p} \frac{1}{\gamma_{lm}^2} \sin(\alpha_l x_j) \cos(\beta_m y_j). \quad (8)$$

III. METHODOLOGY

A. Generative model for the space-charge Hamiltonian prediction

In this work, we propose using a GAN to generate the space-charge Hamiltonian in the transverse plane based on the charge distribution of the beam. Our choice of this particular model is based on experiences of testing a range of machine-learning architectures under the same training conditions, spanning from autoencoders to several GAN variants. With the predicted Hamiltonian, the space-charge forces can be derived by differentiating it. This task can be considered an image-to-image task, as the transverse beam distribution can be converted to a 2D image.

Generative models are a kind of machine learning algorithm designed to solve generative modeling problems. The goal of generative models is to learn the probability distribution of the training data and then generate new samples that agree with the distribution. GAN is one of the most successful approaches to generative modeling problems. A typical GAN consists of two neural networks trained simultaneously, a generator and a discriminator. The generator is trained to generate new data that mimics the target distribution. The discriminator is trained to distinguish between the generator's output and the real data. The discriminator acts as an adversary to the generator, providing feedback to improve the quality of the generated data.

To generate high-fidelity data of the space-charge Hamiltonian, a U-Net architecture with skip connections is used as the generator (see Fig. 1). The U-Net is a type of deep CNN originally designed for biomedical image segmentation, which quickly gained widespread applications in nearly all major imaging modalities. The U-Net has a symmetric autoencoder structure comprising two main components, an encoder and a decoder. The encoder consists of multiple downsampling layers that progressively reduce the spatial resolution of the input data. Each downsampling block includes a 2D convolutional layer followed by instance normalization [34], which stabilizes training and enhances convergence. The decoder mirrors the encoder, using upsampling layers to progressively restore the spatial resolution. To prevent overfitting, dropout layers with a dropout rate of 0.5 are applied within the decoder.

A key feature of the U-Net architecture is the use of skip connections [35], which connect layers from the encoder to the corresponding layers in the decoder. For instance, features from the first downsampling layer are connected to the sixth upsampling layer in Fig. 1. Rather than simply connecting layers in sequence, these skip connections transmit feature maps from earlier layers to deeper layers, allowing the network to better retain essential features and details throughout its depth. This architecture is particularly well suited for tasks that require precise spatial information, such as image segmentation and denoising.

Figure 2 shows the structure of the discriminator, which is a deep CNN with several downsampling layers. Unlike a traditional GAN discriminator that outputs a single probability score for the entire image, we use a PatchGAN design [36] in the discriminator. This patch-based discriminator evaluates small patches in the image instead of the entire image. Each patch is classified as real or fake, allowing the model to focus on local patterns.

B. Data preparation and model training

To capture the behaviors of the space-charge effects under different conditions, we synthesize 10,000 different

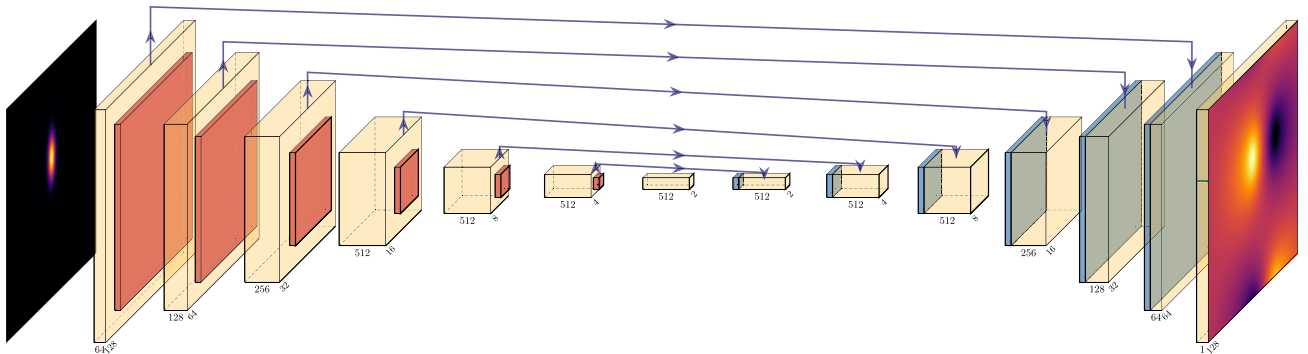


FIG. 1. Schematic diagram of the generator. The generator is a U-Net that consists of seven downsampling layers and seven upsampling layers. The spatial resolution of the image is reduced in the downsampling layers and restored in the upsampling layers.

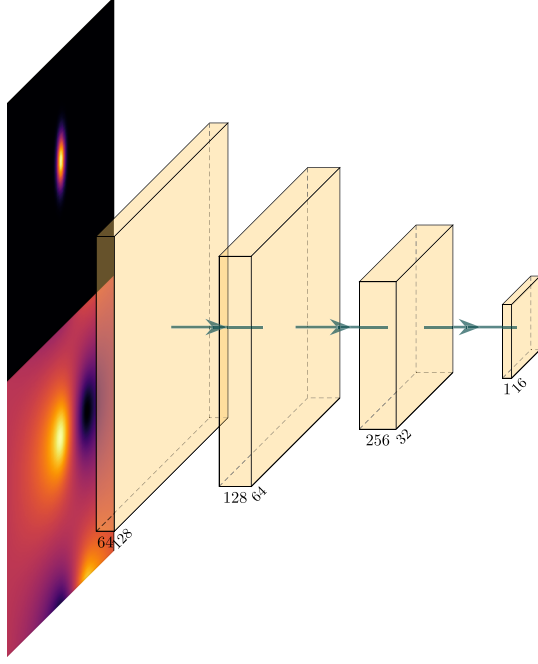


FIG. 2. Schematic diagram of the discriminator. The input images are divided into 16×16 small patches. Each patch is evaluated by the convolutional neural network as real or fake.

beam distributions in $x-y$ space, including Gaussian beams, multi-Gaussian beams, rotated Gaussian beams, core-halo beams, and irregular beams. For Gaussian beams, the standard deviations σ are sampled uniformly up to 2 mm, with centroid offsets in both x and y up to 5 mm. The multi-Gaussian beams are weighted combinations of multiple single Gaussian beams with the relative weights for each of them sampled uniformly. The core-halo beams are

mixtures of a narrow core Gaussian with a broad halo Gaussian, with the core-halo ratio chosen randomly and the core always dominating. The irregular beams are generated from a standard Gaussian perturbed by random spatial distortions and skewness. Each beam contains 1 million particles, which is a challenging number for traditional collective effect simulations. A total of 90% of the datasets are randomly selected as training data, while 10% of them are used for testing.

For the convenience of data processing, the transverse coordinates of all particles are confined to the range of -10 to 10 mm, which is sufficiently large for most high-intensity accelerators. Within the 10×10 mm rectangular range, the space is divided into 128×128 grids for the calculation of the 2D histogram of the beam distribution. The 128×128 histogram is taken as the input image of the machine learning model. The number of grids is chosen to balance model accuracy and efficiency. While increasing the number of grids may enhance accuracy, it also leads to a significant decrease in efficiency.

For the purpose of model prediction, we also generate 128×128 virtual particles positioned at the center of each grid. While these virtual particles do not contribute to the actual space charge, we can calculate the space-charge Hamiltonian they experience due to the real beam by the physical model introduced in Sec. II. The space-charge Hamiltonian of the virtual particles is converted to an image with 128×128 pixels, which is the target image of the generator. The space-charge Hamiltonian values of the real particles can be derived by interpolating to the image.

The generator and the discriminator are trained simultaneously with the training data. Two loss functions, $\mathcal{L}_G$ and $\mathcal{L}_D$, are defined for the generator and discriminator, respectively,

$$\mathcal{L}_G = -\frac{1}{N} \sum_{i=1}^N \{y_i \log[D(x_i, G(x_i))] + (1 - y_i) \log[1 - D(x_i, G(x_i))]\} + \lambda \frac{1}{N} \sum_{i=1}^N |G(x_i) - Y_i| \quad (9)$$

$$\mathcal{L}_D = -\frac{1}{N} \sum_{i=1}^N [y_i \log(D(x_i, y_i))] - \frac{1}{N} \sum_{i=1}^N \{(1 - y_i) \log[1 - D(x_i, G(x_i))]\} \quad (10)$$

where x_i is the input data of the i th training sample, y_i represents its label (1 for real samples and 0 for fake samples), $G(x)$ represents the fake data generated by the generator based on the input x , $D(x, y)$ represents the discriminator's output, and Y represents the real target image.

The first term in $\mathcal{L}_G$ is the adversarial loss that encourages the generator to produce images as realistic as possible to fool the discriminator, and the second term in $\mathcal{L}_G$ is the L_1 loss measuring the pixelwise similarity between the

target and generated images. Based on a systematic parameter search, the coefficient of the L_1 loss λ is set to be 1. The first and second terms in $\mathcal{L}_D$ represent the loss function for evaluating the real and fake samples, respectively, which encourage the discriminator to output 1 for real images and 0 for fake images.

The training of a GAN is known to be challenging due to the unique adversarial dynamics between the generator and the discriminator. When one of the two neural networks progresses faster than the other, the competition between

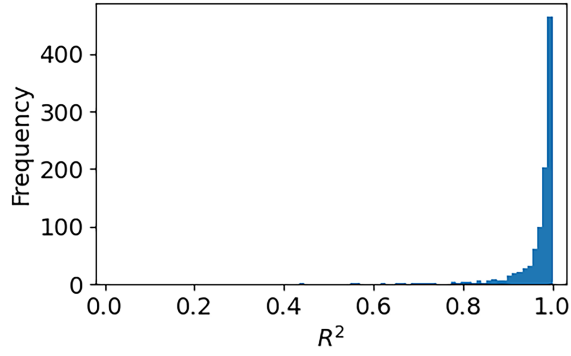


FIG. 3. The coefficient of determination R^2 calculated for all the test data. The x axis is the value of R^2 and y axis is the number of samples.

them becomes imbalanced, leading to one-sided outcomes. For instance, if the generator gains an advantage, it might start producing a limited set of similar images that can reliably fool the discriminator. Conversely, if the discriminator becomes too strong, it may perfectly identify all generated images as fake, leaving the generator with minimal gradient information to learn from. In this work, it is empirically found that it is easier to train the discriminator than the generator. To maintain a balanced training dynamic, we set the discriminator's learning rate to be one-tenth of the generator's learning rate. Additionally, rather than training the discriminator at every iteration, we update it only once for every five generator updates. This

approach helps to prevent the discriminator from overpowering the generator, fostering a more stable adversarial process. All neural networks used in this work are constructed and trained with PyTorch [37], a powerful open-source machine learning platform in Python. The training takes nearly 20 h on a NVIDIA A100 Tensor Core GPU.

IV. RESULTS

A. Model evaluation

To evaluate the predictive performance of the machine learning model, we apply the model to the testing data to predict the space-charge Hamiltonian. Figure 3 shows the coefficient of determination, R^2 , for all the test data, which evaluates how well the observed outcomes are predicted by the model. An R^2 of 1 indicates that the model's predictions perfectly fit the data, while an R^2 close to 0 suggests a poor fit. The results indicate that, for most of the test data, the model achieves high accuracy, with an R^2 over 0.9. This high accuracy demonstrates that the model reliably captures the underlying relationships within the data, providing robust predictions across a variety of test cases.

Despite the overall high accuracy achieved on the test data, a few cases exhibit low R^2 values. These worst-performing cases are typically core-halo distributions having unusually large halos, which make it challenging for the model to extract meaningful information from the input. Nevertheless, such cases represent only a small fraction of

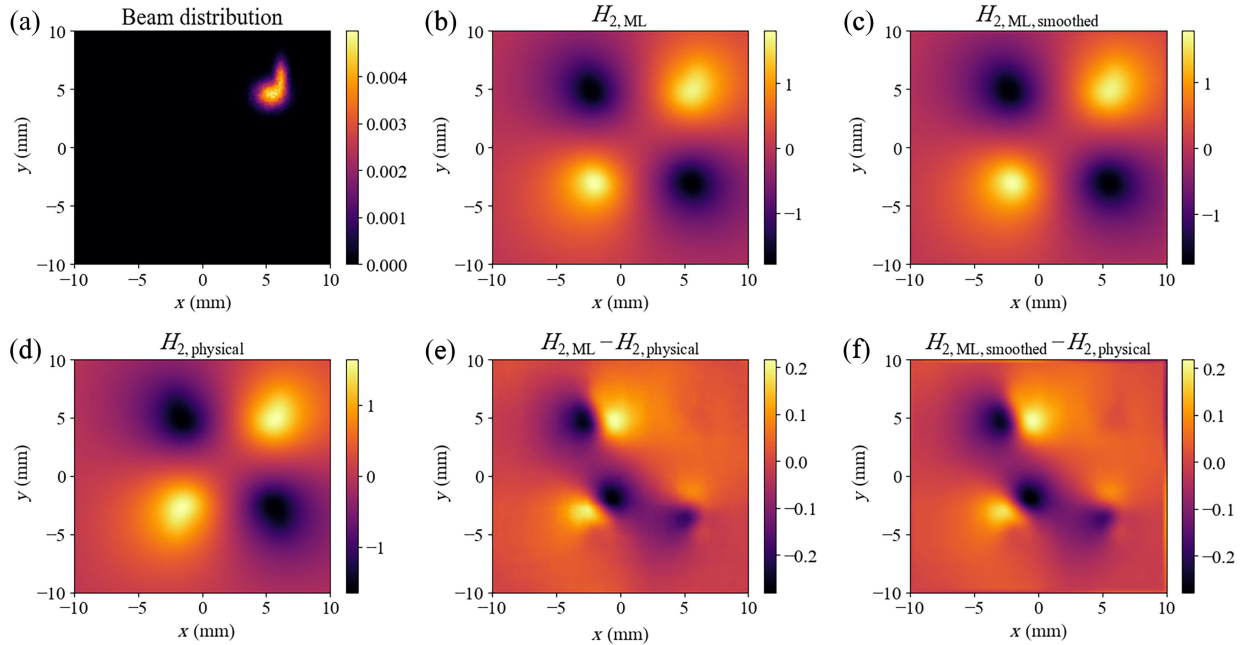


FIG. 4. Beam distribution that is the combination of two different Gaussian beams and the corresponding predicted space-charge Hamiltonian. (a) 2D histogram of beam distribution in $x - y$ plane. (b) Predicted space-charge Hamiltonian values on all virtual particles in the 10×10 mm space. (c) Smoothed prediction. (d) Space-charge Hamiltonian calculated by the physical model, i.e., the ground truth in the training of the generative model. (e) Difference between the prediction and the truth. (f) Difference between the smoothed prediction and the truth.

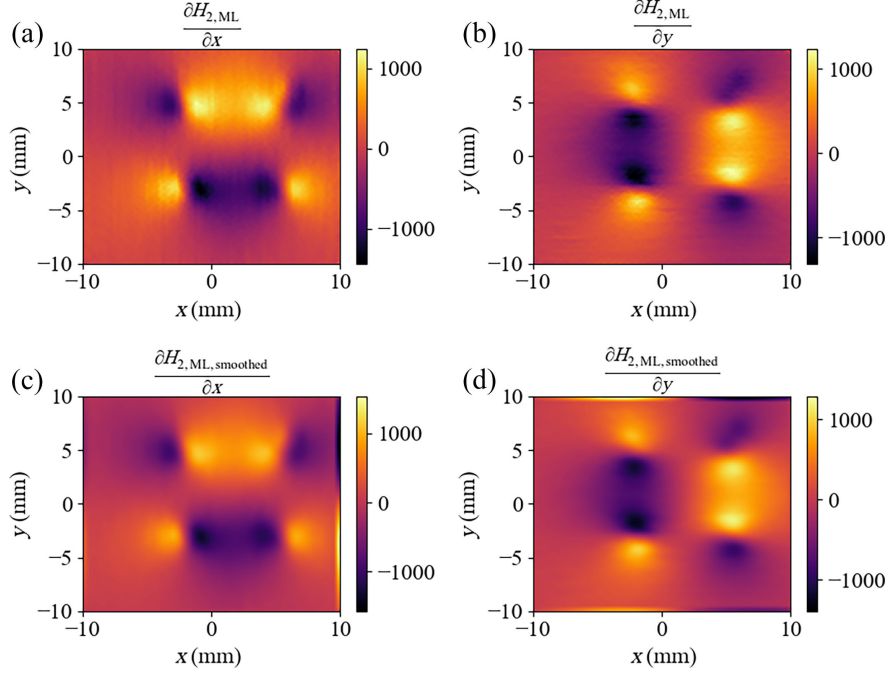


FIG. 5. Derivatives of the predicted space-charge Hamiltonian and derivatives of the smoothed prediction. (a),(b) Derivatives of the prediction with respect to x and y , respectively. (c),(d) Derivatives of the smoothed prediction with respect to x and y , respectively.

the data, and the model's predictive performance remains reliable across the majority of beam distributions.

To validate the performance of the generative model, a new validation sample is created as shown in Fig. 4(a), which is the combination of two Gaussian beams whose centers are each offset by 6 mm, just beyond the 5 mm maximum used during training. This beam distribution is generated solely for testing purposes and may be unrealistic. Figure 4(b) presents the predicted space-charge Hamiltonian values for all virtual particles located at the center of each grid, with the generalized perveance K normalized to 1. When compared to the real space-charge Hamiltonian calculated using the symplectic physical model [see Fig. 4(d)], the predicted Hamiltonian demonstrates reasonable accuracy in the beam region. While larger errors are observed in the area where no particles are present, the prediction remains accurate within the beam region, with errors staying below 5%.

To ensure the symplectic condition, it is essential to build a differentiable function of the Hamiltonian, instead of the discrete predictions on the virtual particles given by the generative model. In each integration step, the space-charge force is approximated by an external field described by this Hamiltonian and all particles experience the same field. Here we fit a cubic spline function to the predicted space-charge Hamiltonian values on the virtual particles. This spline representation of the Hamiltonian is given by

$$H_2(x, y) = \sum_{i=1}^{N_x} \sum_{j=1}^{N_y} c_{i,j} B_i(x) B_j(y), \quad (11)$$

where N_x and N_y denote the number of basis functions B in each dimension, and $c_{i,j}$ are the fitting coefficients.

The derivatives of the Hamiltonian, shown in Figs. 5(a) and 5(b), are computed by differentiating the spline expression in Eq. (11). However, this process shows noticeable noise, primarily due to small discontinuous fluctuations in the predicted Hamiltonian values at the grid points, resulting in irregular and noisy derivatives. To reduce this noise, Gaussian kernel smoothing [38] is applied to the predicted space-charge Hamiltonian values. The smoothed Hamiltonian is calculated by convolving the original Hamiltonian with a 2D Gaussian kernel, as expressed by

$$Y_{\text{smoothed}} = Y_{\text{original}} * \text{kernel}_{2D} \quad (12)$$

where kernel_{2D} is the 2D Gaussian kernel and $*$ denotes the 2D convolution operation.

The smoothed prediction is shown in Fig. 4(c). While the differences between the smoothed and original predictions are small, the effect of smoothing is clearly visible in the derivatives. In Figs. 5(c) and 5(d), it is observed that the smoothing process significantly improves the smoothness of the derivatives, reducing the irregularities and noise presented in the unsmoothed results. Although some artifact errors are introduced at the boundaries of the 10×10 mm space due to the smoothing process, the beam region itself is improved with greater consistency.

By differentiating the smoothed Hamiltonian, we calculate the space-charge forces applied to each particle in the

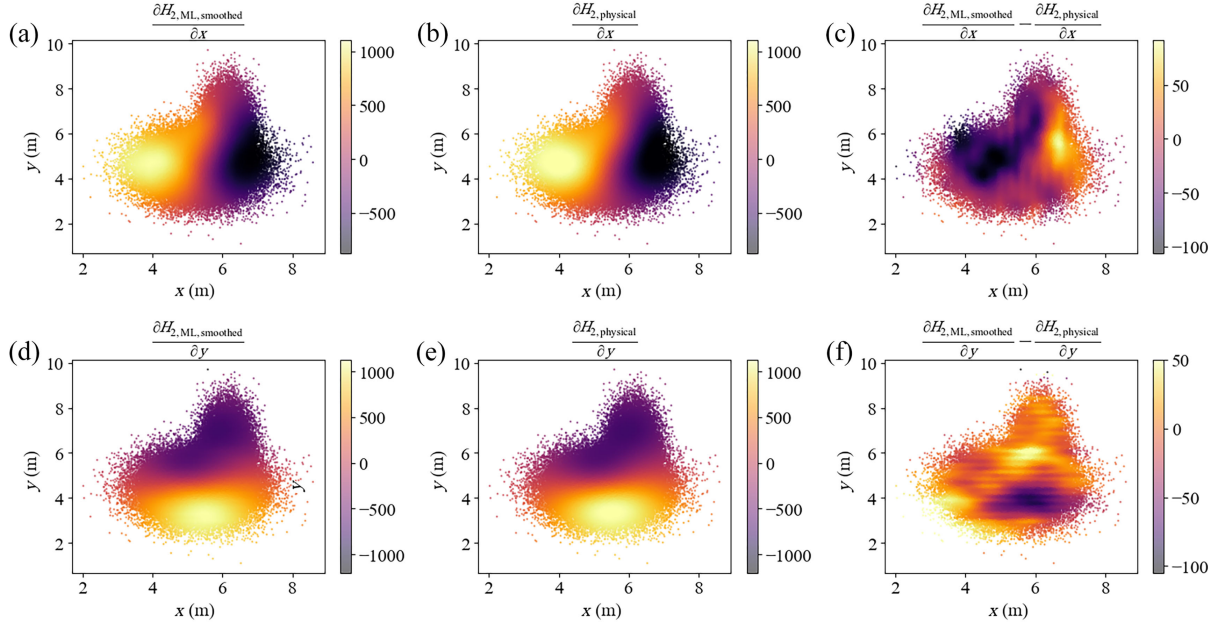


FIG. 6. The space-charge kicks applied on a million particles within the beam shown in Fig. 4(a). The upper row shows the horizontal forces and the lower row is the vertical forces. The first column from left to right presents the results obtained from the generative model with Gaussian kernel smoothing, the second column presents the results of the spectral approach calculated with JuTrack, and the third column is the difference between the two approaches.

beam, which are essential for determining the one-step symplectic kicks. Figure 6 illustrates the space-charge forces acting on one million particles in the double-Gaussian beam. The results obtained using the generative model, after smoothing, align closely with those calculated using the physical model in JuTrack [29], with a maximum error of less than 10%. This agreement demonstrates the model's effectiveness in capturing the complicated space-charge effects within the beam, suggesting the reliability of the generative model in approximating space-charge kicks for beam dynamics simulations.

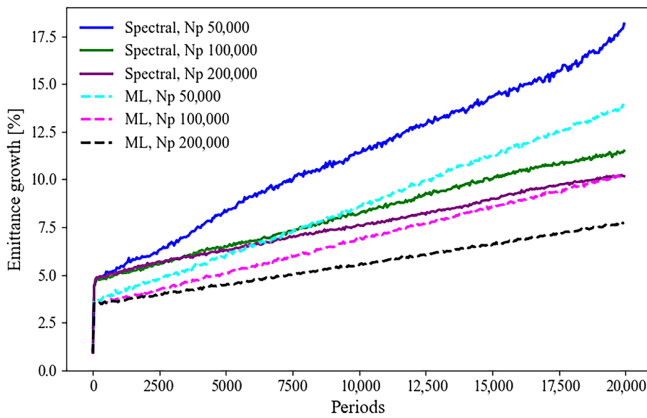


FIG. 7. Emittance growth caused by space-charge effects calculated by the physical model and the generative model. Np represents number of macroparticles implemented.

B. Space-charge-induced emittance growth

Emittance growth caused by space-charge effects can degrade beam quality and even result in particle loss in high-intensity accelerators. Accurately simulating this growth over long-term tracking is challenging, as numerical emittance growth can still occur, even in self-consistent symplectic models. To investigate space-charge-induced emittance growth, we simulate a 1 GeV kinetic energy proton beam transported through periodic FODO cells using JuTrack. The transverse beam distribution is Gaussian with a normalized emittance of 1 mm mrad, and the beam current is set to 100 A. The FODO cell structure consists of drift spaces (D1, D2, and D3) and quadrupoles (Q1 and Q2). The lengths of the drift spaces are 0.2 m for D1, 0.4 m for D2, and 0.2 m for D3. Q1 and Q2 are quadrupoles with opposing focusing strengths, each with a length of 0.1 m.

Figure 7 illustrates the emittance growth, $(\frac{\epsilon_x}{\epsilon_{x0}} \frac{\epsilon_y}{\epsilon_{y0}} - 1)$, over 20,000 periods, as calculated by the generative model using different numbers of macroparticles. For comparison, the physical model is also used to compute space-charge-induced emittance growth, with a spectral mode of 15 and the same number of macroparticles.

The results in Fig. 7 reveal a strong dependence of emittance growth on the number of macroparticles, with more macroparticles leading to reduced emittance growth, implying that both methods are subject to Monte Carlo sampling errors. Initially, both models show a rapid increase in emittance due to the redistribution of the mismatched beam, while the generative model exhibits a slightly lower rate of increase compared to the physical

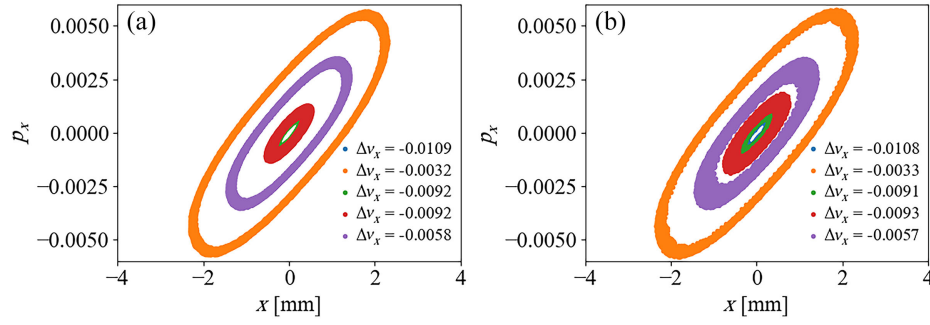


FIG. 8. Comparison of trajectories of several particles in $x - p_x$ space within 20,000 periodic FODO cells. (a),(b) Trajectories distorted by space-charge effects calculated by the physical model and the generative model, respectively. $\Delta\nu_x$ represents tune shift caused by space-charge effects.

model. This discrepancy is mainly due to the difference in the space-charge Hamiltonian given by the two models. Once the beam is matched, the rapid growth of emittance stops, and the subsequent emittance growth rates are close for both models, as evidenced by the nearly parallel slopes of the curves for each macroparticle count. This alignment demonstrates that the generative model and the physical model agree well in predicting long-term behaviors.

Figure 8 shows several trajectories over 20,000 periodic FODO cells in $x - p_x$ space. The tunes are calculated based on the last 1000 periods of trajectories. Here the strength of the second quadrupole is slightly reduced to avoid the linear resonance. Compared to the baseline case without space charge, the trajectories clearly reveal distortions introduced by the space-charge forces. Although the generative model's trajectories exhibit differences relative to the physical model, both models predict close space-charge-induced tune shifts, implying that the dominant nonlinearities are accurately retained by the generative model.

C. Efficiency of the model

To evaluate the efficiency of the generative model, we simulate the space-charge effects for beams with varying

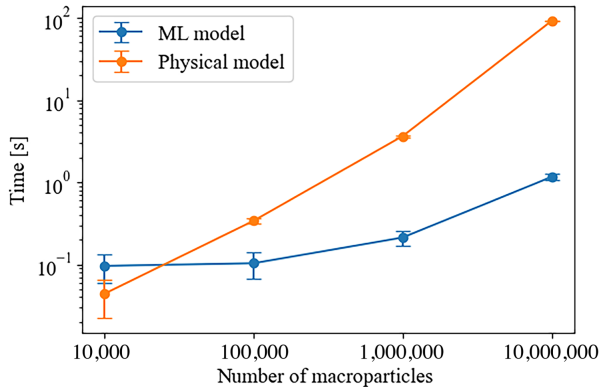


FIG. 9. Computational time required by the generative model and the spectral model for each space-charge kick calculation.

numbers of particles and measure the computational time required by the generative model and the spectral model. All the computations are conducted on a personal laptop with a moderate CPU without using GPU capabilities. Note that here we only consider the time required for the field solver, and the time required for the particle pusher is not considered, as this process is typically the same for different methods. The computational time for calculating a single space-charge kick using the generative method includes the time for 2D histogram calculation, model prediction, Gaussian kernel smoothing, and differentiation of the smoothed Hamiltonian to calculate the space-charge forces on all particles.

As shown in Fig. 9, the generative model takes slightly more time than the spectral model when the number of particles is only 10,000. This is due to the overhead associated with generating the predicted Hamiltonian and performing the interpolation, which dominates the computational cost at lower particle counts. When the number of particles exceeds 100,000, the overhead in the generative model becomes negligible, and both the spectral model and generative model exhibit a linear increase in the computational time as the number of particles grows. Compared to the spectral model, the generative model remains consistently faster by more than an order of magnitude.

This performance is attributed to the generative model's ability to compute the Hamiltonian over the entire grid in a single forward pass, avoiding the computationally intensive particle-to-particle interactions typically required for space-charge calculations. In contrast, the spectral method's computational cost grows linearly with the number of particles, making it less efficient for large-scale simulations. In addition, the efficiency of the generative model can be further enhanced with advanced GPU devices, as the model built on PyTorch is naturally designed for GPU acceleration. The results highlight the scalability and practicality of the generative model for high-intensity beam simulations, where large particle numbers are necessary for accurate representation of space-charge effects.

V. CONCLUSION

In this work, we introduced a symplectic generative machine learning model to efficiently simulate transverse space-charge effects. By using a U-Net architecture within a GAN framework, our model predicts the transverse space-charge Hamiltonian based on the particle distribution and derives the corresponding symplectic transfer map via differentiation. The proposed model preserves the symplectic nature of beam dynamics, reducing nonphysical errors for long-term simulations.

The results demonstrate that the model achieves high accuracy in predicting the space-charge Hamiltonian, with an R^2 exceeding 0.9 across most test cases. By applying Gaussian kernel smoothing to the predicted Hamiltonian, the gradient noise in space-charge forces is significantly reduced, ensuring consistent results for space-charge kicks. Furthermore, the generative model provides an order-of-magnitude improvement in computational efficiency compared to the spectral method, particularly for simulations involving a large number of particles.

We demonstrate a practical application of the generative model by integrating it into JuTrack, a Julia-based autodifferentiable accelerator modeling code. This model effectively preserves symplecticity in the simulation, resulting in similar growth rates of transverse emittance compared to the physical model in long-term tracking.

The symplectic machine learning model represents an efficient alternative in the simulation of collective effects such as space-charge effects. Its high accuracy, computational efficiency, and ability to preserve phase-space structure make it a valuable tool for the simulation of high-intensity accelerators. Future work could explore extending the model to three-dimensional space-charge simulations and incorporating additional physical effects, such as beam-beam interactions and wakefield forces.

ACKNOWLEDGMENTS

This work is 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.

DATA AVAILABILITY

This code for implementing this model is available in JuTrack [29]. The code for generating the training data and the code for model training is available at [39].

-
- [1] Marco Durante, Alexander Golubev, Woo-Yoon Park, and Christina Trautmann, Applied nuclear physics at the new high-energy particle accelerator facilities, *Phys. Rep.* **800**, 1 (2019).

- [2] Vladimir Shiltsev and Frank Zimmermann, Modern and future colliders, *Rev. Mod. Phys.* **93**, 015006 (2021).
- [3] Kenneth R. Long, Donatella Lucchesi, Mark A. Palmer, Nadia Pastrone, Daniel Schulte, and V. Shiltsev, Muon colliders to expand frontiers of particle physics, *Nat. Phys.* **17**, 289 (2021).
- [4] Valeriia N. Starovoitova, Lali Tchelidze, and Douglas P. Wells, Production of medical radioisotopes with linear accelerators, *Appl. Radiat. Isot.* **85**, 39 (2014).
- [5] Robert W. Garnett, Overview of accelerators with potential use in homeland security, *Phys. Procedia* **66**, 196 (2015).
- [6] P. A. Bystrov, A. V. Gordeev, A. Yu Kolokolova, M. A. Zavyalov, N. V. Ilyukhina, A. A. Molin, Y. S. Pavlov, S. P. Polyakova, A. V. Prokopenko, and V. P. Filippovich, Prospects of electron beam irradiation to ensure microbiological safety of food products, *Phys. At. Nucl.* **81**, 1526 (2018).
- [7] Ji Qiang, Long-term simulation of space-charge effects, *Nucl. Instrum. Methods Phys. Res., Sect. A* **918**, 1 (2019).
- [8] Ji Qiang, Robert D. Ryne, Salman Habib, and Viktor Decyk, An object-oriented parallel particle-in-cell code for beam dynamics simulation in linear accelerators, *J. Comput. Phys.* **163**, 434 (2000).
- [9] Andrei Shishlo, Sarah Cousineau, Jeffrey Holmes, and Timofey Gorlov, The particle accelerator simulation code PyORBIT, *Procedia Comput. Sci.* **51**, 1272 (2015).
- [10] Dhanoj Gupta, Raj Singh, Ryan Ringle, Catherine R. Nicoloff, Igor Rahinov, Oded Heber, and Daniel Zajfman, Particle-in-cell techniques for the study of space charge effects in an electrostatic ion beam trap, *Phys. Rev. E* **104**, 065202 (2021).
- [11] Ji Qiang, Symplectic multiparticle tracking model for self-consistent space-charge simulation, *Phys. Rev. Accel. Beams* **20**, 014203 (2017).
- [12] Auralee Edelen, Nicole Neveu, Matthias Frey, Yannick Huber, Christopher Mayes, and Andreas Adelmann, Machine learning for orders of magnitude speedup in multi-objective optimization of particle accelerator systems, *Phys. Rev. Accel. Beams* **23**, 044601 (2020).
- [13] Jinyu Wan, Paul Chu, and Yi Jiao, Neural network-based multiobjective optimization algorithm for nonlinear beam dynamics, *Phys. Rev. Accel. Beams* **23**, 081601 (2020).
- [14] Andrei Ivanov and Ilya Agapov, Physics-based deep neural networks for beam dynamics in charged particle accelerators, *Phys. Rev. Accel. Beams* **23**, 074601 (2020).
- [15] Jinyu Wan and Yi Jiao, Machine learning enabled fast evaluation of dynamic aperture for storage ring accelerators, *New J. Phys.* **24**, 063030 (2022).
- [16] Ryan Sandberg, Remi Lehe, Chad Mitchell, Marco Garten, Andrew Myers, Ji Qiang, Jean-Luc Vay, and Axel Huebl, Synthesizing particle-in-cell simulations through learning and GPU computing for hybrid particle accelerator beamlines, in *Proceedings of the Platform for Advanced Scientific Computing Conference* (Association for Computing Machinery, New York, NY, 2024), pp. 1–11.
- [17] Auralee Edelen and Christopher Mayes, Neural network solver for coherent synchrotron radiation wakefield calculations in accelerator-based charged particle beams, in *Proceedings of the 13th International Particle Accelerator Conference, IPAC2022, Bangkok, Thailand* (JACoW, Geneva, Switzerland, 2022), pp. 2261–2264.

- [18] R. Robles, W. Lou, C. Mayes, R. Roussel, and A. Edelen, Efficient computation of two-dimensional coherent synchrotron radiation with neural networks, in *Proceedings of the 14th International Particle Accelerator Conference, IPAC2023, Venice, Italy* (JACoW, Geneva, Switzerland, 2023), pp. 2892–2895.
- [19] Christopher Leon, Petr M. Anisimov, Nikolai Yampolsky, and Alexander Scheinker, Using convolutional neural networks to accelerate 3D coherent synchrotron radiation computations, [arXiv:2503.09551](https://arxiv.org/abs/2503.09551).
- [20] Jan Kaiser, Chenran Xu, Annika Eichler, and Andrea Santamaria Garcia, Bridging the gap between machine learning and particle accelerator physics with high-speed, differentiable simulations, *Phys. Rev. Accel. Beams* **27**, 054601 (2024).
- [21] C. K. Huang, Q. Tang, Y. K. Batygin, O. Beznosov, J. Burby, A. Kim, S. Kurennoy, T. Kwan, and H. N. Rakotoarivelo, Symplectic neural surrogate models for beam dynamics, *J. Phys. Conf. Ser.* **2687**, 062026 (2024).
- [22] Christopher Leon and Alexander Scheinker, Physics-constrained machine learning for electrodynamics without gauge ambiguity based on Fourier transformed Maxwell's equations, *Sci. Rep.* **14**, 14809 (2024).
- [23] Alexander Scheinker and Reemu Pokharel, Physics-constrained 3D convolutional neural networks for electrodynamics, *APL Mach. Learn.* **1**, 026109 (2023).
- [24] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, Generative adversarial networks, *Commun. ACM* **63**, 139 (2020).
- [25] Jinyu Wan and Yi Jiao, Machine learning-based direct solver for one-to-many problems on temporal shaping of relativistic electron beams, *Front. Phys.* **17**, 64601 (2022).
- [26] Ryan Roussel, Juan Pablo Gonzalez-Aguilera, Eric Wisniewski, Alexander Ody, Wanming Liu, John Power, Young-Kee Kim, and Auralee Edelen, Efficient six-dimensional phase space reconstructions from experimental measurements using generative machine learning, *Phys. Rev. Accel. Beams* **27**, 094601 (2024).
- [27] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, U-Net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention*, in *Proceedings of the 18th international conference on Medical image Computing and Computer-Assisted Intervention, MICCAI-2015, Munich, Germany* (2015), Vol. 9351, pp. 234–241, [10.1007/978-3-319-24574-4_28](https://doi.org/10.1007/978-3-319-24574-4_28).
- [28] Zewen Li, Fan Liu, Wenjie Yang, Shouheng Peng, and Jun Zhou, A survey of convolutional neural networks: Analysis, applications, and prospects, *IEEE Trans. Neural Networks Learn. Syst.* **33**, 6999 (2021).
- [29] Jinyu Wan, Helena Alamprese, Christian Ratcliff, Ji Qiang, and Yue Hao, JuTrack: A Julia package for auto-differentiable accelerator modeling and particle tracking, *Comput. Phys. Commun.* **309**, 109497 (2025).
- [30] Matt Signorelli and David Sagan, SciBmad: A full-featured ecosystem for modern, differentiable accelerator physics simulations, in *Proceedings of the 14th International Computational Accelerator Physics Conference* (2025), <https://indico.gsi.de/event/19249/contributions/82658/attachments/48832/70947/SciBmad.pdf>.
- [31] Ji Qiang, Differentiable self-consistent space-charge simulation for accelerator design, *Phys. Rev. Accel. Beams* **26**, 024601 (2023).
- [32] David Gottlieb and Steven A. Orszag, *Numerical Analysis of Spectral Methods: Theory and Applications* (SIAM, Philadelphia, PA, 1977), [10.1137/1.9781611970425](https://doi.org/10.1137/1.9781611970425).
- [33] Ji Qiang, Symplectic particle-in-cell model for space-charge beam dynamics simulation, *Phys. Rev. Accel. Beams* **21**, 054201 (2018).
- [34] D. Ulyanov, D. Vedaldi, and V. Lempitsky, Instance normalization: The missing ingredient for fast stylization, [arXiv:1607.08022](https://arxiv.org/abs/1607.08022).
- [35] Michal Drozdal, Eugene Vorontsov, Gabriel Chartrand, Samuel Kadoury, and Chris Pal, The importance of skip connections in biomedical image segmentation, in *Proceedings of the International Workshop on Deep Learning in Medical Image Analysis (DLMIA 2016), Athens, Greece* (2016), Vol. 10008, pp. 179–187, [10.1007/978-3-319-46976-8_19](https://doi.org/10.1007/978-3-319-46976-8_19).
- [36] Ugur Demir and Gozde Unal, Patch-based image inpainting with generative adversarial networks, [arXiv:1803.07422](https://arxiv.org/abs/1803.07422).
- [37] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Antiga Luca *et al.*, PyTorch: An imperative style, high-performance deep learning library, in *Proceedings of the 33rd International Conference on Neural Information Processing Systems* (Curran Associates Inc., Red Hook, NY, 2019), Vol. 32.
- [38] Matt P. Wand and M. Chris Jones, *Kernel Smoothing* (CRC Press, Boca Raton, FL, 1994).
- [39] J. Wan, Essential code for a symplectic machine learning-based space-charge effect model, Zenodo, [10.5281/zenodo.15328235](https://doi.org/10.5281/zenodo.15328235) (2025).