GENERATIVE FIELD INVERSION IN A CROSSED-FIELD ION SOURCE

Thomas A. Marks
Department of Aerospace Engineering
University of Michigan
marksta@umich.edu

Alex A. Gorodetsky
Department of Aerospace Engineering
University of Michigan
goroda@umich.edu

ABSTRACT

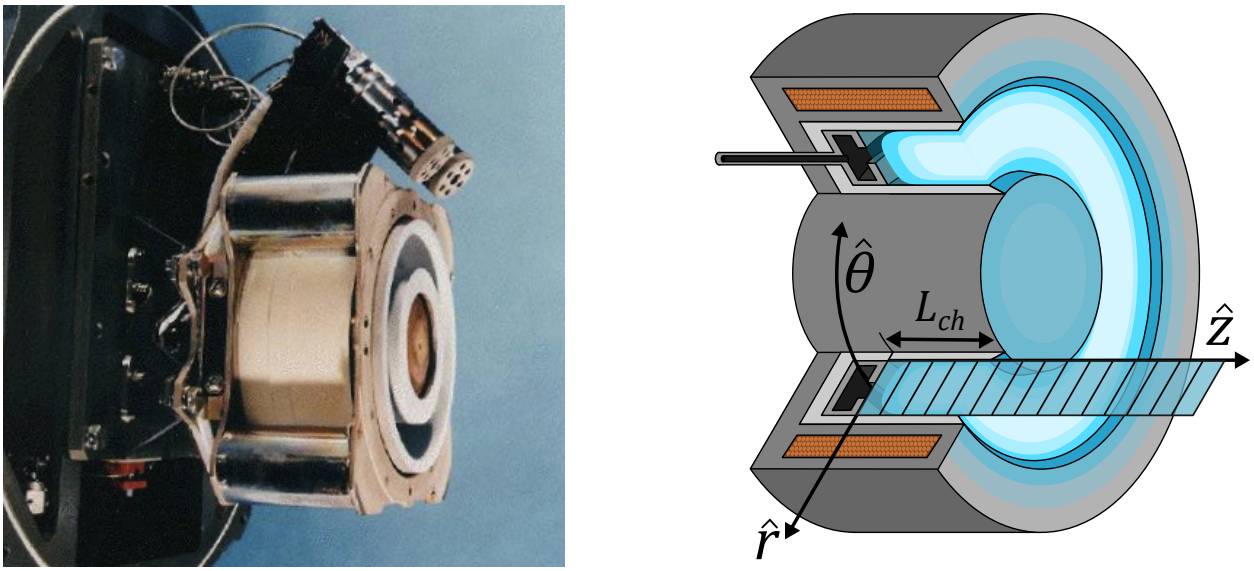
A generative model is used to probabilistically reconstruct the full one-dimensional state of a crossed-field ion source from partial, uncertain observations. The model, a diffusion model, is trained on a dataset of one-dimensional simulations of an SPT-100 Hall effect thruster and sampled by iterative denoising, guided by sparse measurements. It evaluated as a forward surrogate, where it precisely estimates the plasma density, electron temperature, and ion velocity when conditioned on an input anomalous collision frequency profile. The model also matches ground-truth results from Markov chain Monte Carlo (MCMC) when conditioned on observation of the ion velocity. It additionally solves probabilistic inverse problems on real and simulated experimental data, with results that match or exceed existing semi-analytic methods in quality, flexibility, and robustness to noise and missing data fields. Ablations are performed with respect to the amount of data required, the size of the model, the form of the loss function, and the number of steps taken during sampling. The cost of the method is compared to MCMC, and potential applications of the method to data assimilation and model predictive control are outlined.

1 Introduction

Crossed-field ion sources, such as Hall-effect thrusters, Penning discharges, and magnetrons, play host to a range of plasma instabilities which produce enhanced electron transport across magnetic field lines [1]. As this transport cannot presently incorporated self-consistently into engineering simulations, it is commonly modeled as a spatially-varying, time-independent "anomalous collision frequency" specified by the user and calibrated to make simulations match available data [2]. The outputs of these calibrated simulations can be inspected to estimate other unobserved plasma properties. Simulations calibrated in this way are not unique, however, and two simulations that agree well on one observable may disagree strongly on another [3]. Uncertainty quantification (UQ) is thus highly important. Historically, however, model calibration has been largely manual, and UQ has been neglected. In recent years, model calibration approaches based on classical [4] or stochastic [5] optimization have have become available, but these methods either only partially incorporate UQ or are slow due to the reliance on sequential sampling methods like Markov chain Monte Carlo (MCMC). Other techniques attempt to infer unobserved states, including the collision frequency, from measurements by simplifying the governing equations to make them tractable [6, 7, 8, 9]. Obtaining the data required to employ these methods can be challenging, and the simplifications made to aid analysis may impact the validity of the results. There is thus a need for methods which can rapidly infer the anomalous collision frequency and unobserved states from sparse data, while quantifying the uncertainty in the resulting state estimates.

Denoising diffusion models [10] provide a potential solution to these challenges. These generative models are a class of neural network commonly used for image and text generation. Due to their robustness and ability to sample high-dimensional distributions [11], they have also been applied to scientific problems, including the inversion of PDEs and the inference of unknown closure fields [12, 13]. While promising, they have typically been applied to simple PDEs like Darcy Flow [13] and Burgers' equation [14] with only one or two state variables. Additionally, they have not been extensively used to assimilate noisy experimental data.

In this work, we apply diffusion models to probabilistic field inversion in simulations of a Hall-effect thruster. We accomplish this by first generating a large dataset of random input parameters, including random anomalous collision



(a) (b)

Figure 1: (a) Picture of an SPT-100 Hall thruster. (b) Domain of 1-D axial simulations in *HallThruster.jl.*

frequency curves, which we pass through a 1-D fluid simulation of a Hall effect thruster [15]. The simulation inputs and outputs, comprising 17 spatially-varying fields, are arranged into a tensor and used to train a diffusion model. As a result, the model learns the prior distribution of possible plasma states from our database of simulations. We can then sample from this prior distribution, or optionally condition the model on partial observation of one or more fields to sample from the posterior distribution over the unobserved fields. As sampling occurs in parallel, we rapidly obtain many posterior samples which can be used for UQ. The conditioning procedure we employ works at sampling-time, allowing it to solve arbitrary probabilistic forward and inverse problems without re-training the underlying model.

The diffusion-based inference method we demonstrate in this work fills several gaps in existing methods. Unlike the existing semi-analytical techniques used to invert experimental data [6, 8, 9], the proposed method does not rely on the presence of specific fields and gracefully handles sparse and noisy data. Additionally, compared to classical sampling-based inference methods like Markov Chain Monte Carlo (MCMC), the cost of the present method is largely paid offline, during training. Online inference is comparatively rapid, enabling the proposed method to be used in data-assimilation and streaming settings where new data is constantly obtained and inference needs to be performed multiple times. Finally, we note that while we focus on Hall thrusters in this work, this method is largely agnostic to the specific simulation or physical system used to generate training data and can be easily extended to similar systems.

Our paper is organized as follows. In Sec. 2, we describe the 1-D simulations performed, the data generation procedure, as well as the architecture of the diffusion model and the methods for training and sampling it. Next, in Sec. 3, we present the results of our model both as a forward surrogate for the 1-D code and as a tool for solving inverse problems. We evaluate its performance in an experimental setting, conditioning it on real measurements and comparing its outputs to those obtained from two semi-analytic methods. In Sec. 4, we discuss our model's computational cost and use cases and enumerate some remaining challenges before summarizing our findings in Sec. 5. In the Appendix, we evaluate the sampling performance of the diffusion model as a function of data size, model size, training methods, and sampling procedure.

2 Methods

2.1 Data generation

We build a diffusion model for the SPT-100 Hall thruster, shown in Fig. 1a. This is a 1 kW-class Hall thruster developed by Fakel in Russia [16], which has become a standard reference thruster for modeling activities [5, 17]. Despite the relative abundance of experimental data available for this thruster [16, 18], there is still insufficient data for training a diffusion model. Instead, we generate training data using the open-source one-dimensional axial fluid Hall thruster code *HallThruster.jl* [15]. This code aims to captures many of the important physics driving the evolution and

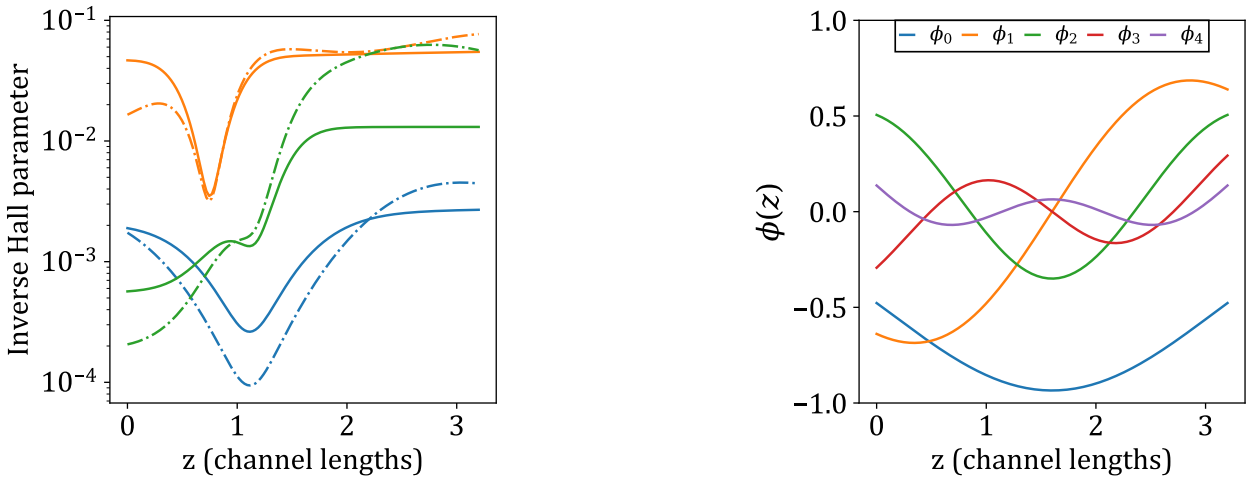
Table 1: Parameters and distributions sampled during data generation, as well as model outputs. The uniform distribution is $\mathcal{U}(a, b)$, while the normal and log-normal distributions are $\mathcal{N}(\mu, \sigma)$ and LN(μ, σ), respectively. The truncated versions of these distributions are then TN(μ, σ, a, b) and TLN(μ, σ, a, b).

Scalar parameters			
Symbol	Description	Units	Prior distribution
$\dot{m}_a$	Anode mass flow rate	$\mathrm{mg\,s^{-1}}$	$\mathcal{U}(3, 6)$
V_d	Discharge voltage	V	$\mathcal{U}(200, 600)$
f_B	Magnetic field scale	-	$\mathcal{U}(0.5, 1.5)$
V_{cc}	Cathode coupling voltage	V	$\mathcal{U}(0, 50)$
u_n	Axial neutral velocity	$\mathrm{m\,s^{-1}}$	TN$(300, 100, 0, \infty)$
f_w	Wall loss scale	-	TN$(1, 0.5, 0, 2)$
Anomalous transport parameters			
c_1	Scale	-	LN$(\log(0.1), \log(5)$
c_2	Center location	-	TN$(1.1, 1/3, 0, 2)$
c_3	Anode-cathode step size	-	$\mathcal{U}(0, 1)$
c_4	Anode-cathode slope	-	$\mathcal{U}(0, 1)$
c_5	Barrier scale	-	TLN$(\log(0.05), \log(10), 0, 1)$
c_6	Barrier width	-	TN$(0.3, 0.5, 0, \infty)$
w_1	Noise weight 1	-	LN$(0, 1)$
w_2	Noise weight 2	-	$\mathcal{N}(0, 1)$
w_3	Noise weight 3	-	$\mathcal{N}(0, 1)$
w_4	Noise weight 4	-	$\mathcal{N}(0, 1)$
w_5	Noise weight 5	-	$\mathcal{N}(0, 1)$
Output fields			
B	Magnetic field strength	G	-
ν_e	Total electron collision frequency	$\mathrm{s^{-1}}$	-
ν_{an}	Anomalous electron collision frequency	$\mathrm{s^{-1}}$	-
n_n	Neutral number density	$\mathrm{m^{-3}}$	-
n_e	Electron number density	$\mathrm{m^{-3}}$	-
u_e	Axial electron velocity	$\mathrm{m\,s^{-1}}$	-
$n_{i,\{1,2,3\}}$	Ion number density (Z = 1, 2, 3)	$\mathrm{m^{-3}}$	-
$u_{i,\{1,2,3\}}$	Axial ion velocity (Z = 1, 2, 3)	$\mathrm{m\,s^{-1}}$	-
ϕ	Electrostatic potential	V	-
E	Axial electric field	$\mathrm{V\,m^{-1}}$	-
T_e	Electron temperature	eV	-
p_e	Electron pressure	$\mathrm{eV\,m^{-3}}$	-
∇p_e	Electron pressure gradient	$\mathrm{eV\,m^{-4}}$	-

acceleration of a Hall thruster discharge plasma while remaining fast enough for parameter estimation and inference workflows [19]. This speed also makes it ideal for generating a large amount of training data for machine learning tasks.

Tab. 1 provides the inputs and outputs of *HallThruster.jl* used in this work, and Fig 1b shows the simulation domain. The domain extends from the anode boundary at $z = 0$ to the cathode boundary at $z = 8\,\mathrm{cm}$ and includes $M = 128$ cells. At the anode boundary, neutral xenon atoms are injected with a mass flow rate $\dot{m}_a$ at a velocity of u_n. The electrostatic potential is set to the discharge voltage, V_d, at the anode and the cathode coupling voltage, V_{cc}, at the cathode. We fix the magnetic field profile to that of the SPT-100, and we allow its magnitude to vary using a scale parameter f_B. Finally, the code includes a scale factor, f_w, which modifies the effective edge-to-center density ratio used to compute electron and ion wall losses. We collect these scalar input variables $[\dot{m}_a, V_d, f_B, V_{cc}, u_n, f_w]$ into a vector $\mathbf{v}$ and assign each a prior distribution (also listed in Tab. 1), which we sample randomly for each example in our training dataset.

In addition to these variables, *HallThruster.jl* also requires the specification of an anomalous electron collision frequency curve, $\nu_{an}(z)$. Generating the necessary data thus requires us to sample ν_{an} curves randomly. Many parametric models of $\nu_{an}(z)$ exist and are commonly employed in Hall thruster simulations [20, 21, 3, 22]. Typically, these follow the Bohm scaling, such that the inverse anomalous Hall parameter, $\Omega_{an}^{-1} = \nu_{an}/\omega_{ce}$ is a function of z only. These range from two-region step-out profiles [20] to arbitrarily-complex piecewise-linear or piecewise-logarithmic



(a) (b)

Figure 2: (a) Three sample curves of the inverse anomalous Hall parameter ($\Omega_{an}^{-1} = \nu_{an}/\omega_{ce}$) from the six-parameter anomalous transport model in Eq 1, with (dashed lines) and without (solid lines) the addition of correlated Gaussian noise. (b) First five eigenfunctions of the squared exponential kernel with $l = L_{ch}$ on a domain of [0, 80] mm with a kernel length of 25 mm.

profiles [4, 23]. In this work, we begin with a flexible six parameter base model which expresses Ω_{an}^{-1} as a product of a logistic curve $f(z)$ and an inverted Gaussian $g(z)$:

$$\Omega_{an}^{-1}(\hat{z}) = \nu_{an}/\omega_{ce} = c_1 f(\hat{z}) g(\hat{z}), \tag{1}$$

$$f(\hat{z}) = 1 - c_3 \left(1 - \frac{1}{1 + \exp\left[-\frac{c_4 \hat{z}}{1-c_4}\right]}\right), \tag{2}$$

$$g(\hat{z}) = 1 - (1 - c_5) \exp\left[-\frac{\hat{z}}{c_6}^2\right], \tag{3}$$

$$\hat{z} = z/c_2 - 1. \tag{4}$$

$$\tag{5}$$

The form of the base model allows both for a rise in ν_{an} moving from the anode to the near-plume region [3] as well as a region of reduced transport, which in Hall thruster models serves to ensure that a strong accelerating electric field emerges [24]. This model also recovers the classic two-region Bohm transport model [20, 21] when $c_4 = 1$ and $c_5 = 0$. As with the scalar parameters, we generate ν_{an}(z) curves by specifying prior distributions of c_1–c_6 which we sample randomly.

While the base model is flexible, it remains limited in its expressive power. To further increase the range of possible $\nu_{an}(z)$ curves, we perturb the base model with correlated gaussian noise with a squared exponential covariance kernel, *i.e.*,

$$k(z, z') = \beta^2 \exp\left[\frac{1}{2}\frac{(z - z')^2}{l^2}\right]. \tag{6}$$

Here, β and is the standard deviation of the noise and l is the correlation length scale, which we set to 1.0 and L_{ch} respectively. We generate a realization of the noise, $\hat{X}$, by sampling a multivariate normal distribution with a covariance matrix $K \in \mathbb{R}^{M \times M}$, $K_{i,j} = k(z_i, z_j)$. We add the noise to $\log(\Omega_{an}^{-1})$ to permit variations across orders of magnitude, as

$$\log\left(\tilde{\Omega}_{an}^{-1}\right) = \log\left(\Omega_{an}^{-1}\right) + \hat{X}, \qquad \text{where } \hat{X} \sim \mathcal{N}(0, K). \tag{7}$$

In Fig. 2a, we plot several sampled anomalous collision frequency curves, both with and without the added Gaussian noise. Adding correlated Gaussian noise transforms our anomalous collision frequency model from parametric to non-parametric. This is not a challenge for diffusion models, but poses difficulties for the parametric inference methods

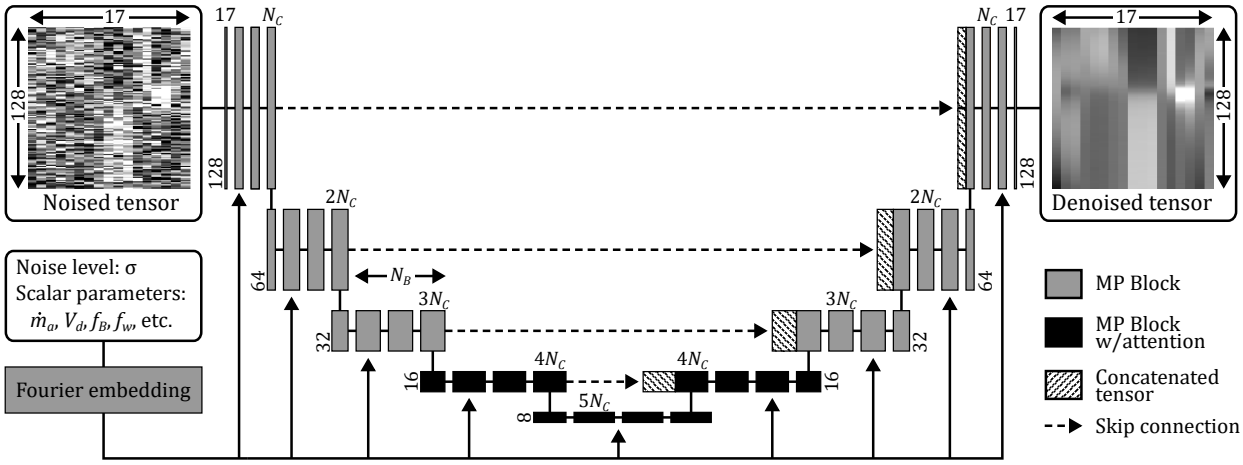


Figure 3: Schematic diagram of the network architecture employed in this work.

such as MCMC to which we will compare our approach. To this end, we parameterize the Gaussian noise by performing a Karhunen-Lòeve expansion of the squared exponential kernel in terms of its eigenvalues λ_i and eigenfunctions φ_i [25]:

$$K\varphi_i = \lambda_i \varphi_i. \tag{8}$$

Summing the first p eigenvalues produces a realization of $\hat{X}$,

$$\hat{X} \approx \sum_{i=1}^{p} w_i \sqrt{\lambda_i} \varphi_i, \tag{9}$$

where each weight parameter w_i is a random variable following a standard normal distribution. The original Gaussian process is recovered as $p \to \infty$, but the eigenvalues decay quickly, allowing us to truncate the expansion and obtain a finite parametric model for the noise. This gives a parametric model with eleven parameters, c_1–c_6 and w_1–w_5, which we list in Tab. 1. When running MCMC, we retain just the first five eigenfunctions ($p = 5$), which we show in Fig. 2b. We use the full non-parametric form of the noise when generating training data for the diffusion model.

To generate data, we sample a random set of scalar inputs from Tab. 1 as well as a random anomalous collision frequency curve using the non-parametric form of the procedure described above. At each set of inputs, we run *HallThruster.jl* with three ion charge states on a grid of $M = 128$ cells for one millisecond of simulation time, which takes 3 seconds on a single thread of an AMD Ryzen 9 9900X. We generate 2^{21} training examples in parallel batches of 24. After data generation is complete, we average each simulation's outputs in time over the final half-millisecond of the simulation. We save $C = 17$ fields from each simulation; these are also listed in Tab. 1 and normalize the simulation outputs per-field to have zero mean and a standard deviation of 0.5 following Karras *et al* [11]. These outputs are then laid out into 2-D tensor of dimension $M \times C$ and saved along with the scalar parameters $\mathbf{v}$, which are normalized with the same method.

2.2 Model architecture and training

We employ a modified 1-D version of the EDM2 architecture from Ref [26] as our diffusion model. The model architecture, shown in Fig. 3, is a U-net consisting of four encoder levels, a mid-level/bottleneck level, and four decoder levels. At the start of the network, the number of channels in increased from C to N_C. The encoder levels down-sample the tensor, reducing the resolution by two and increase the number of channels by N_C. The decoder levels up-sample the tensor, concatenate the tensor from the same resolution on the decoder side along the channel dimension, and decrease the number of channels. Each level (encoder or decoder) contains a configurable number of residual blocks, N_B. For our base model, we take $N_B = 3$ and $N_C = 64$; we investigate the effect of changing these values in Appendix A. The model takes as input a tensor of the 17 fields listed in Tab. 1 and outputs a denoised version of the same tensor. We additionally inform the model of both the vector of scalar inputs $\mathbf{v}$ associated with a specific training example and the noise level σ using Fourier position embeddings.

At each training step, we add uncorrelated Gaussian noise with a random standard deviation σ to a batch of training data examples $\overline{x} \in \mathbb{R}^{B \times C \times M}$, with B being the batch size. The noise level is sampled per training example from $\log \sigma \sim$

$\mathcal{N}(p_{mean}, p_{std})$, with $p_{mean} = -1.2$ and $p_{std} = 1.2$, following Ref. [11]. This produces the noised tensor $x(\sigma)$. Then, the diffusion model, $D(x; \sigma, \mathbf{v})$, predicts the original noise-free tensor. The loss is computed as the reconstruction loss $||D(x; \sigma, \mathbf{v}) - \overline{x}||_2^2$, multiplied by a noise-dependent weight function, $\lambda(\sigma) = (\sigma^2 + \sigma_{data}^2)/(\sigma\,\sigma_{data})^2$ [11]. To improve the smoothness of our solutions, we additionally incorporate first- and second-derivative penalties in our loss function. The modified training loss is given below, with f_D indicating a tuneable scale for the derivative loss terms.

$$\mathcal{L} = \lambda(\sigma)\left[\|D(x;\sigma,\mathbf{v}) - y\|_2^2 + f_D\,\|\Delta_z\,(D(x;\sigma,\mathbf{v}) - y)\|_2^2 + f_D^2\,\left\|\Delta_z^2\,(D(x;\sigma,\mathbf{v}) - y)\right\|_2^2\right]. \tag{10}$$

$$x(\sigma) = \overline{x} + \xi, \quad \xi \sim \mathcal{N}(0, \sigma^2\,\mathbf{I}^{B\times C\times M}) \tag{11}$$

In this work, we take $f_D = 1$; we investigate the impact of this choice in Appendix A. We train the model using the Adam optimizer for 1024 epochs, or 2^{31} training examples. On an Nvidia H100 GPU, training takes 1 day, 22 hours at a batch size of $B = 8192$ and a learning rate of 1.2×10^{-3}. Meanwhile, on a consumer-grade Nvidia Gefore RTX 3060, it takes 4 days, 8 hours at a batch size of 2048 and a learning rate of 3×10^{-4}.

2.3 Sampling

To sample the model, we begin with a sample of pure Gaussian noise and integrate backward toward zero noise along trajectories defined by a probability flow ODE:

$$\frac{dx}{d\sigma} = -\sigma\nabla_x \log p(x;\sigma,v) = -\frac{1}{\sigma}(D(x;\sigma,\mathbf{v}) - x). \tag{12}$$

Here, $\nabla_x \log p(x; \sigma, v)$ is the *score function*, a vector field which transports samples of Gaussian noise toward the data distribution. In this formulation, sampling is implicitly conditioned on some choice of the scalar parameter vector $\mathbf{v}$. Purely unconditional samples can be generated by drawing random instances of $\mathbf{v}$ from the prior distributions in Tab. 1 at the start of sampling.

To condition on partial observations, we sample from the posterior distribution, $p(x \mid y; \sigma, \mathbf{v})$ instead of the prior distribution, $p(x; \sigma, \mathbf{v})$. Applying Bayes' rule, the sampling ODE for the posterior becomes

$$\frac{dx}{d\sigma} = -\sigma\left(\nabla_x \log p(x;\sigma,\mathbf{v}) - \nabla_x \log p(y \mid x;\sigma,\mathbf{v})\right), \tag{13}$$

where $\log p(y \mid x; \sigma, \mathbf{v})$ is the likelihood. As the noisy sample x is conditionally-independent of the measurement y with respect to the clean sample, x_0, the likelihood can be expressed by marginalizing over x_0 as

$$p(y \mid x;\sigma,\mathbf{v}) = \int_{x_0} p(x_0 \mid x)p(y \mid x_0)\,\mathrm{d}x_0. \tag{14}$$

We model the observations, y, as generated from the clean image prediction, x_0, according the forward noise model,

$$y = A(x_0) + \eta, \quad \eta \sim \mathcal{N}(0, \gamma^2\,\mathbf{I}^m), \tag{15}$$

where $A : \mathbb{R}^{B\times C\times M} \to \mathbb{R}^m$ is a measurement operator and η is the measurement noise with standard deviation γ. A acts on normalized quantities so γ is given relative to the normalized range of that quantity. This formulation gives $p(y \mid x_0) = \mathcal{N}(A(x_0) \mid y, \gamma^2\,\mathbf{I}^m)$. However, $p(x_0 \mid x)$ is intractable, so we follow Chung *et al.* in using a single-point likelihood estimate at the mean clean sample prediction for Eq 14, or

$$\mathbb{E}_{x_0}[p(y \mid x)] \approx p(y \mid \mathbb{E}[x_0]). \tag{16}$$

Combining this with the noise model in Eq. 15, we obtain an expression for the likelihood score,

$$\nabla_x \log p(y \mid x;\sigma,\mathbf{v}) \approx -\frac{1}{\gamma^2}\nabla_x\,\|y - A(D(x;\sigma,\mathbf{v}))\|_2^2\,, \tag{17}$$

where we compute the gradient with respect to x is with automatic differentiation. In practice, choosing γ from the measurement noise alone gives inappropriately high guidance during the early phases of sampling, when noise levels are high and the log-likelihood of the clean sample with respect to the data is very low. This leads to sampling instabilities, which we remedy by scaling γ with the current noise level as

$$\gamma^2 = \gamma_{data}^2(1 + r^2), \tag{18}$$

where γ_{data} is the "true" measurement noise and $r^2 = \sigma^2/(1 + \sigma^2)$ is an additional variance derived from Ref. [27] which approximates the variance of $p(x_0 \mid x)$. In addition, following Refs. [13, 28, 29], we apply a guidance strength

Algorithm 1 Conditional sampling with two-stage Runge-Kutta scheme with parameter α.
Derived from Refs. [11, 28, 13].

procedure DERIVATIVE(x, σ, $D(x;\sigma,\mathbf{v})$)
  $\hat{x}_0 \leftarrow D(x;\sigma,\mathbf{v})$ ▷ Obtain clean sample prediction
  $s \leftarrow (x-\hat{x}_0)/\sigma^2$ ▷ Evaluate score (Eq. 12)
  return $-\sigma s$, $\hat{x}_0$ ▷ Return $\partial x/\partial\sigma$ (Eq. 13) and clean sample
end procedure

procedure CONDITIONALSAMPLER($D(x;\sigma,\mathbf{v})$, $\sigma_{i\in\{0,\dots,S\}}$, $\mathbf{v}$, $A(x)$, y, α, γ)
  sample $x_S \sim \mathcal{N}(0, \sigma_S^2\,\mathbf{I})$ ▷ Generate initial noise sample
  for $i \leftarrow S$ to 1 **do**
    $\Delta\sigma \leftarrow \sigma_{i-1} - \sigma_i$
    $f_0, \hat{x}_0 \leftarrow$ DERIVATIVE(x_{i-1}, σ_i, D)
    $x'_{i-1} \leftarrow x_i + \alpha f_0$ ▷ First Runge-Kutta stage
    $f_1, \hat{x}_0 \leftarrow$ DERIVATIVE$(x'_{i-1}, \sigma_i + \alpha\Delta\sigma, D)$
    $x''_{i-1} \leftarrow x_i + \Delta\sigma\left(\left(1-\frac{1}{2\alpha}\right) f_0 + \frac{1}{2\alpha} f_1\right)$ ▷ Second Runge-Kutta stage
    $x_{i-1} \leftarrow x''_{i-1} + \zeta\,\nabla_x \|y - A(x_0)\|_2^2$ ▷ Apply observation guidance
  end for
  return x_0 ▷ Return clean sample at $\sigma = 0$
end procedure

ζ at each step instead of directly integrating the measurement likelihood as written. This guidance strength can be expressed in terms of γ as

$$\zeta = \frac{1}{C^2\gamma^2(1+r^2)}. \tag{19}$$

We take $C = 40$, which ensures that samples have appropriate uncertainties for a given γ.

With the ODE defined, we conditionally sample from the diffusion model by integrating Eq. 13 from a starting noise level σ_{max} to $\sigma = 0$. Algorithm 1 gives our sampling procedure for an arbitrary two-stage Runge-Kutta integrator, parameterized by coefficient α. Here, $\alpha = 1$ corresponds to the Heun integrator used in Ref. [11]. Our sampling procedure instead uses the midpoint method ($\alpha = 1/2$), which was slightly more stable than Heun's method during testing. The noise steps σ_i are chosen following Ref. [11], and decay with noise level following a degree-7 polynomial. Unless otherwise specified, we take $S = 2048$ steps for each generation in this work, and all presented results derive from a batch of 256 samples. These sampling parameters are summarized along with the architectural and training parameters in Tab. 2.

3 Results

3.1 Forward surrogate

We first demonstrate our network's ability to capture key physics of our system by acting as a forward surrogate. To do this, we condition the diffusion model on the anomalous collision frequency and scalar parameters of a reference simulation. The parameters of the reference simulation, reported in Tab. **??** were chosen to represent a realistic operating condition for an SPT-100 Hall thruster with representative plasma parameters.

In Fig. 4, we show the distributions of diffusion model predictions for four plasma properties after conditioning on the inputs to the reference simulation with a sampling noise of $\gamma = 0.025$. This value was the smallest that yielded stable sampling. The ion velocity and electron temperature are captured accurately with low uncertainty, and the median and 50% credible intervals of the plasma density coincide with the reference simulation. We also observe that the imposed constraints on the anomalous collision frequency are well-obeyed during sampling, with the sampled curves closely following the reference. The 95th percentile of diffusion samples shows a small amount of uncertainty in the plasma density both near the anode and outside of the discharge channel. As we will see later, the plasma density is highly sensitive to small changes in the anomalous collision frequency. The small but finite value of γ used may have thus manifested as a larger amount of uncertainty in n_e. Taken together, however, these results showcase the ability of the diffusion model to accurately model the physical relationships between simulation fields and demonstrate the success of the proposed posterior sampling procedure.

Table 2: Data, architecture, training, and sampling parameters used in this work.

Data parameters		
Name	Symbol	Value
Training dataset size	N_D	2^{21}
Data channels	C	17
Resolution	M	128
Number of scalar parameters	P	6
Data standard deviation	σ_{data}	0.5
Architecture parameters		
Base channels	N_C	64
Blocks per level	N_B	3
Self-attention max resolution	-	16
Self-attention heads	-	64
Training parameters		
Epochs	-	1024
Batch size (1 x Nvidia H100)	B	8192
Batch size (1 x Nvidia Geforce RTX 3060)	B	2048
Learning rate (1 x Nvidia H100)	α	1.2×10^{-3}
Learning rate (1 x Nvidia Geforce RTX 3060)	α	3×10^{-4}
Log noise distribution mean	p_{mean}	-1.2
Log noise distribution std	p_{std}	1.2
Derivative loss weight	f_D	1.0
Sampling parameters		
Sampling steps	S	2048
Number of samples	N_S	256
Maximum noise level	σ_{max}	80
Minimum noise level	σ_{min}	0.002
Measurement standard deviation	γ	1.0

Table 3: Operational and calibration parameters of the reference simulation.

$\dot{m}_a$ (kg/s)	V_d (V)	f_B	V_{cc}	u_n (m/s)	f_w
5×10^{-6}	300	1.0	0.0	150.0	1.0

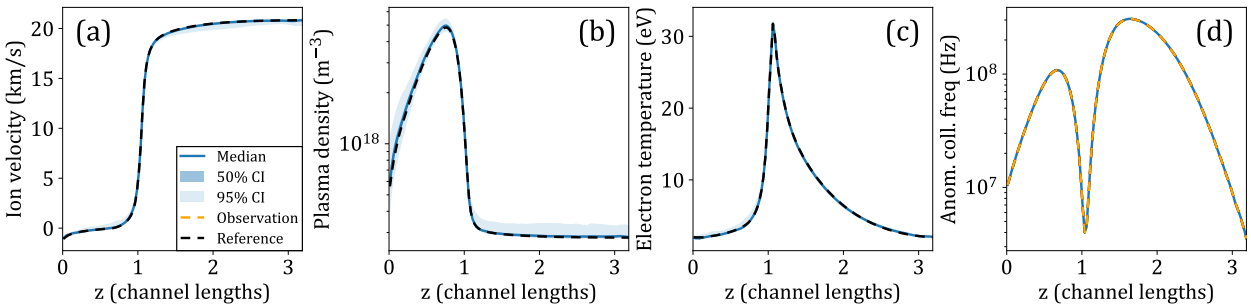


Figure 4: Diffusion model predictions (256 samples) of (a) ion velocity, (b) plasma density, (c) electron temperature, (d) anomalous collision frequency conditioned on the anomalous collision frequency and parameters of the reference simulation. The reference simulation outputs are indicated with dashed black lines, and orange dashed lines highlight the observed quantity (here, ν_{an}). The diffusion model successfully acts as a forward surrogate with low variance.

3.2 Probabilistic inference problem

In this section, we compare the posterior predictive distributions obtained by the diffusion model to those from an Markov chain Monte Carlo (MCMC) sampling procedure, which serves as "ground truth". Here, the task is to infer the distribution of the full plasma state given only the singly-charged ion velocity curve from the reference simulation, which we denote u_{ref}. We formulate the MCMC inverse problem using the same parameters as in Tab. 1, with the addition of the five Gaussian noise weight parameters w_1–w_5 discussed in Sec. 2.1 to make the anomalous transport curve generation parametric. We fix the scalar parameters $\mathbf{v}$ to those of the reference simulation (Tab. 3), giving us an 11-dimensional parameter vector θ. The posterior distribution used in MCMC is obtained as the product of the prior distributions in Tab. 1 and a likelihood derived assuming a Gaussian error with standard deviation γ between u_{ref} and the model output $u_{i,1}(z_j, \theta)$, or

$$\log p(d \mid \theta) = -\frac{n}{2}\log(2\pi\gamma^2) - \sum_{j=0}^{n} \frac{1}{2\gamma^2}\left(u_{i,1}(z_j,\theta) - u_{ref}(z_j)\right)^2. \tag{20}$$

We take $\gamma = 500\,\mathrm{m\,s^{-1}}$ for MCMC sampling, which represents a typical experimental error in ion velocimetry in $E \times B$ sources [18]. Due to the high-dimensional parameter space, a large number of samples are needed for MCMC to converge. We draw 100,000 samples from the posterior using the Delayed Rejection Adaptive Metropolis (DRAM) algorithm [30] and discard the first half as burn-in; this takes about five days on an Apple M4 max chip.

The diffusion model is conditioned on the same ion velocity curve and scalar parameters as the MCMC simulation We use a normalized $\gamma = 0.028$ for the ion velocity noise, which is 500 m/s divided by 17,400 km/s, the normalization constant for singly-charged ion velocities. Fig. 5 compares the posterior predictive distributions of $u_{i,1}$, n_e, T_e, and $\Omega_e^{-1} = \nu_e/\omega_{ce}$ from MCMC sampling to those of the diffusion model. The two distributions agree quantitatively and qualitatively across most of the axial domain. Within the minimal variation in the ion velocity, both the diffusion model and MCMC predict variations of over two orders of magnitude in the near-anode plasma density, indicating that ion velocity alone is not sufficient to constrain this field. The electron temperature curve is better-constrained, with the largest uncertainty occurring in the peak value. This ranges from 30–49 eV in the MCMC results and 30-45 eV in the diffusion results. The inverse electron Hall parameter curve exhibits the largest differences between the two, with the diffusion model predicting higher values at all locations. As has been previously observed [3], knowledge of the ion velocity profile is not sufficient to narrow down the value of the electron collision frequency outside of the location and depth of the minimum; the 95% credible intervals for both cases span more than order of magnitude at the left and right sides of the domain. The differences between the methods can likely be attributed to the approximations made to sample the posterior in Sec. 2.3, approximation error in the diffusion model itself, and the parametric versus non-parametric transport models used.

The quality of the diffusion model's approximation to the posterior distribution depends on the training and sampling parameters employed. Appendix A explores the effect of these parameters quantitatively, finding that the number of sampling steps taken and the amount of data used most strongly affect the quality of the approximation. Whether the approximation error is acceptable depends heavily on the computational cost and convenience of each method. We compare the cost of the present method to MCMC in Sec. 4.1. While the specifics depend on the details of each method, we find that for problems like ours, diffusion models provide large online time savings over MCMC when solving inverse problems, which becomes advantageous when solving such problems repeatedly.

3.3 Comparison to experimental techniques

3.3.1 Method of Perez Luna et al.

Perez-Luna *et al.* [6] use a simplified 1-D Boltzmann equation for singly-charged ions to compute the electric field and ionization rate by taking moments of ion velocity distribution functions (IVDFs) obtained with laser-induced fluorescence (LIF). Defining the normalized 1D velocity distribution function $g(v)$ such that $\int_{-\infty}^{\infty} g(v) = 1$, the m-th moment of $g(v)$, u_m, and the ratio of successive moments, w_m, are given by

$$u_m = \int_{-\infty}^{\infty} v^m g(v)\,\mathrm{d}v, \tag{21}$$

$$w_m = u_m/u_{m-1}. \tag{22}$$

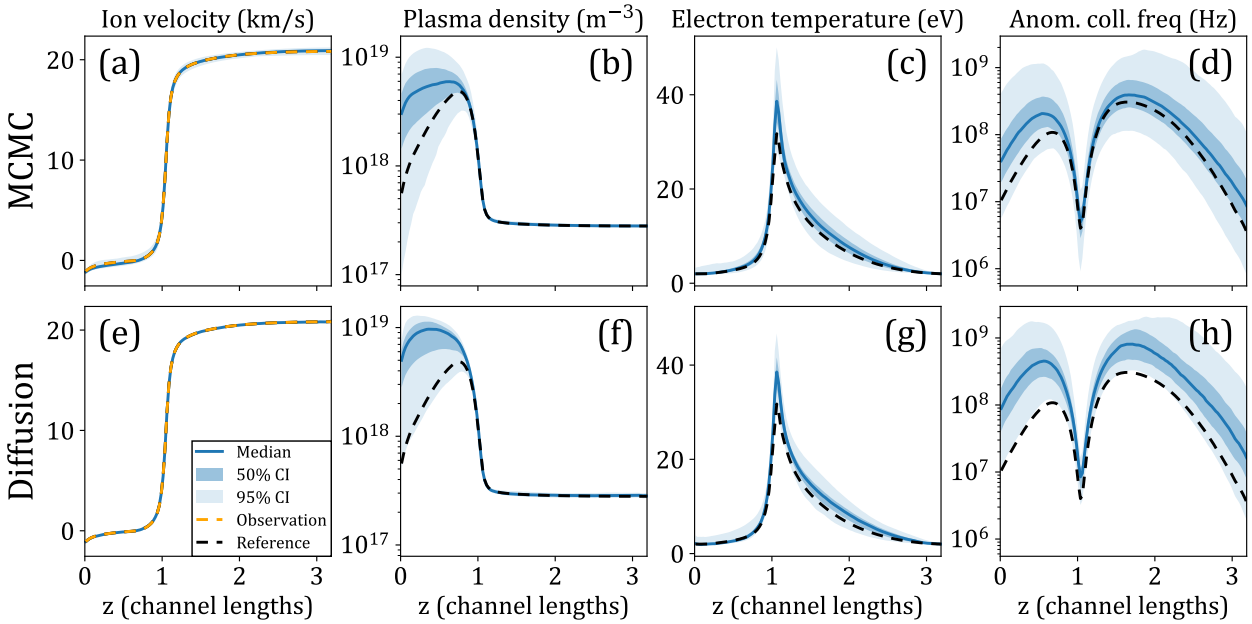


Figure 5: Posterior predictive distributions from 50,000 MCMC samples (top row) compared to 256 samples from the diffusion model (bottow row). The diffusion model approximates the MCMC posterior predictive distributions well.

The axial acceleration a, axial electric field E, and ionization frequency ν_{iz} are then

$$a = \frac{w_1 w_2}{2w_1 - w_3} \frac{\partial w_3}{\partial z}, \tag{23}$$

$$E = \frac{M}{e} a, \tag{24}$$

$$\nu_{iz} = \frac{a}{w_2} - \frac{w_1}{w_2} \frac{\partial w_2}{\partial z}. \tag{25}$$

We evaluate both the diffusion model and Perez-Luna technique on experimental measurements, using use Macdonald-Tenenbaum *et al.*'s LIF measurements of the SPT-100 IVDFs at a background pressure of 35 µTorr [18]. We condition the diffusion model on the most probable velocity of the IVDF. This quantity is less affected by charge exchange collisions and other effects that produce low-energy ions and reduce the mean velocity. As *HallThruster.jl* does not model these processes and includes only a single ion velocity per charge state, this choice gives a more direct comparison. We then compute the ionization frequency from the diffusion model's predicted electron temperatures and neutral and ion densities, using ionization rate coefficients from *HallThruster.jl*. For the scalar parameters, we condition on the $V_d = 300\,\mathrm{V}$, $\dot{m}_a = 5.16\,\mathrm{mg\,s^{-1}}$, $V_{cc} = 30.0$, and $f_B = 1.0$. The values of V_d and $\dot{m}_a$ derive directly from Refs. [18], while V_{cc} for the SPT-100 at this pressure is approximated based on Ref. [31] and $f_B = 1$ indicates that the magnetic field is at its nominal value. As u_n and f_w are not known, we randomly choose unique values from the prior distributions in Tab. 1 for each sample, in effect marginalizing our predictions over these parameters. In Fig. 6, we compare the electric field and ionization rate from the diffusion model to those of the Perez-Luna technique. We additionally compare these results to a third method in which E is computed directly from the mean velocity, neglecting ionization, as

$$E = \frac{M}{e} u \frac{du}{dz}. \tag{26}$$

The diffusion model agrees well with the method of Perez-Luna *et al.* The electric fields agree particularly well, with the diffusion model-predicted peak field differing from the experimentally-derived value by less than 3 kV / m (25.5 vs 22.5 kV/m, respectively). These two methods agree more closely to each other than they do to the simplified method of Eq. 26, which predicts a value of just 18 kV/m. The diffusion model and experimental methods disagree more on the ionization frequency, with the method of Perez-Luna *et al.* predicting a peak ionization frequency 3 times that of the diffusion model. Outside of this peak, from $z/L_{ch} = 1$ to $z/L_{ch} = 1.8$, the methods agree to within uncertainty. Past this point, they diverge, with the diffusion model predicting an ionization rate which continues to decline and the method of Perez-Luna *et al.* predicting that it flattens out at around 10^{-4} Hz. These points are well outside of the

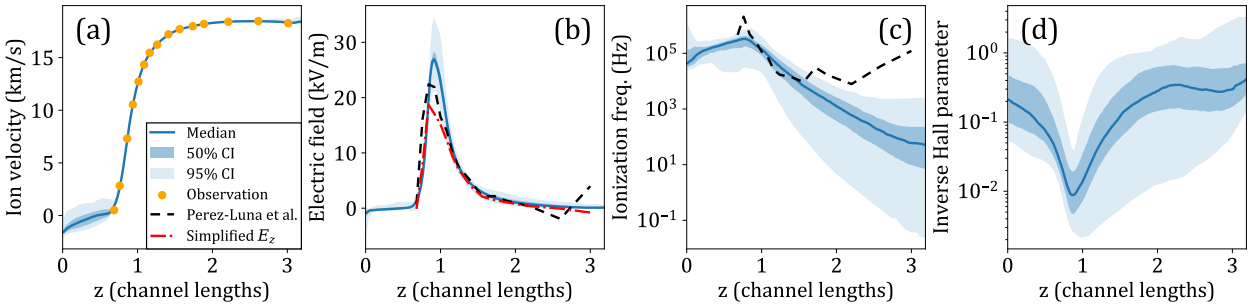


Figure 6: Diffusion model outputs of (a) singly-charged ion velocity, (b) electric field, (c) ionization rate, and (d) inverse electron Hall parameter when conditioned on LIF data from Refs. [18] (observation). Results obtained using the technique of Perez-Luna *et al.* [6] are indicated in black dashed lines, while the simplified electric field obtained from Eq. 26 is in red. The electric field of the diffusion model agrees well with that of Perez-Luna's method.

discharge channel of the SPT-100, so the 1-D assumptions underlying both our method and that of Perez-Luna *et al.* begin to break down. It is likely that neither method produces an accurate estimate in this region. Finally, we obtain an estimate of the inverse electron Hall parameter, which has a shape consistent with the ion velocity curve but varies over an order of magnitude. The relatively large variation likely stems from the marginalization over u_n, and f_w, each of which would impact what the anomalous collision frequency would need to be to produce the ion velocity profile from experiment. These results demonstrate that the diffusion model can be successfully conditioned on sparse experimental data, despite being only trained on simulations, and provide estimates of fields beyond what can be calculated using standard experimental techniques.

3.3.2 Roberts' method

Roberts and Jorns inverted a simplified 1D model of a Hall thruster semi-analytically to obtain the inverse electron Hall parameter, Ω_e^{-1}. The one-dimensional fluid electron Ohm's law gives [9]:

$$\Omega_e^{-1} = \frac{\nu_{an} + \nu_{class}}{\omega_{ce}} - \frac{u_{ez}}{u_{e,\theta}} \approx -\frac{u_{ez}}{\frac{E}{B} + \frac{1}{en_eB}\frac{\partial}{\partial z} n_e T_e} \tag{27}$$

Here, u_{ez} and $u_{e,\theta}$ are the electron axial and azimuthal velocities, respectively, and ν_{class} is the classical electron collision frequency.. In the magnetized limit where $\Omega_e \gg 1$, $u_{e\theta}$ can be approximated as the sum of the $E \times B$ and diamagnetic drifts, which are the first and second terms in the denominator of Eq. 27, respectively. Armed with this expression, Ω_e^{-1} can be calculated given E, T_e, n_e, and u_{ez}. Roberts and Jorns applied incoherent Thomson scattering (ITS) to non-invasively measure the electron velocity distribution function (EVDF), obtaining n_e and T_e from the zeroth and second moments of the EVDF, respectively. Next, they used LIF to measure $u_{iz,1}$, and then applied the method of Perez-Luna *et al.* to get E. Finally, they determined u_{ez} by integrating a system of ordinary differential equations derived from one-dimensional current conservation:

$$\partial_z J_{ez} = -en_e \sum_{Z=0}^{2} \sum_{Z'=Z+1}^{3} (Z' - Z)\, n_Z K_{(Z \to Z')}(T_e) \tag{28}$$

$$\partial_z J_Z = Zen_e \left[\sum_{Z'=0}^{Z-1} n_{Z'} K_{(Z' \to Z)}(T_e) - \sum_{Z'=Z+1}^{3} n_Z K_{(Z \to Z')}(T_e) \right] \tag{29}$$

$$\partial_z n_0 = -\frac{n_e n_0}{u_n} \sum_{Z=1}^{3} K_{(0 \to Z)}(T_e). \tag{30}$$

In the above, $J_{ez} = -en_e u_{ez}$ is the electron axial current density, J_Z is the current density of ion species with charge Z, for $Z > 1$, n_0 is the density of neutral atoms ($Z = 0$), and $K_{Z \to Z'}(T_e)$ is the ionization rate coefficient for the production of particles of charge state Z' from those of charge Z, evaluated at a given electron temperature. Roberts holds the axial neutral velocity, u_n, constant, which matches the approximation made *HallThruster.jl*. They integrate the above equations upstream from $z = z_{end}$, with with boundary conditions given by

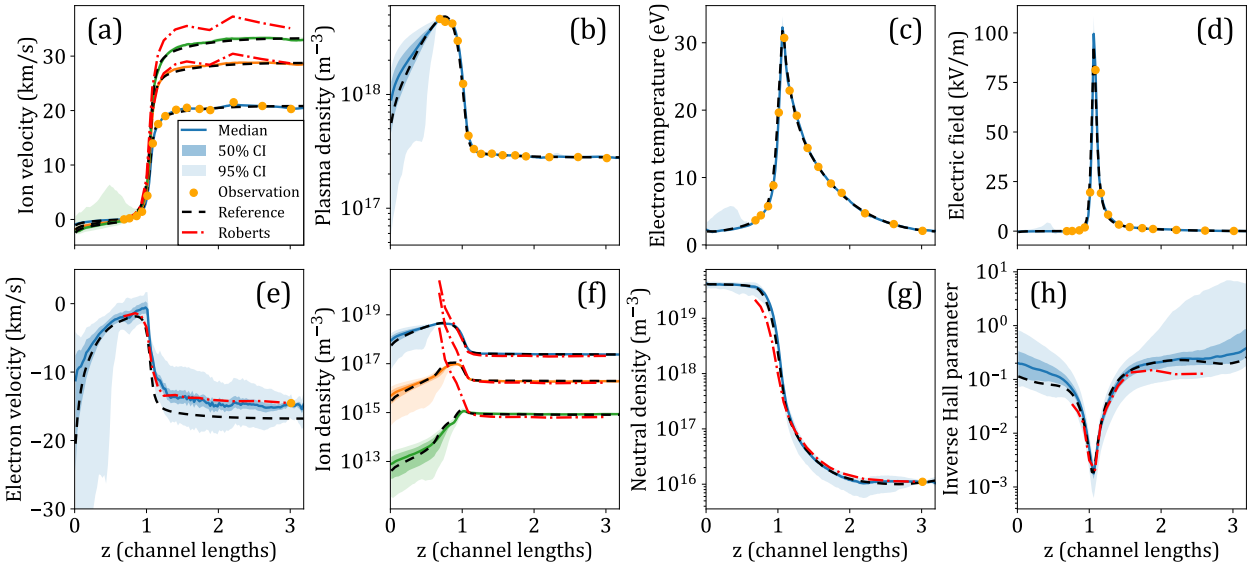


Figure 7: Comparison of the method of Roberts and Jorns (red) to diffusion model outputs (blue) and the reference simulation (black). Conditioning is performed on simulated measurements from laser-induced fluorescence and Thomson scattering (orange markers). Diffusion model predictions for singly, doubly, and triply charged ion densities and velocities are colored blue, orange, and green, respectively. The diffusion model more closely agrees with the reference for all quantities, and successfully extrapolates to the anode.

$$J_i(z_{end}) = e[n_e u_{iz,1}](z_{end}) \sum_Z \omega_Z/\sqrt{Z}, \tag{31}$$

$$J_e(z_{end}) = J_i(z_{end})(1/\eta_b - 1), \tag{32}$$

$$J_Z(z_{end}) = \omega_Z J_i(z_{end}), \tag{33}$$

$$n_n(z_{end}) = \frac{J_i(z_{end})}{e u_n}(1/\eta_m - 1). \tag{34}$$

Here, ω_Z is the fraction of the total ion current J_i carried by ions of charge state Z; $\eta_m = \dot{m}_i/\dot{m}$ is the mass utilization efficiency, or the ratio of the flow rate of ions leaving the thruster to neutrals injected at the anode; and $\eta_b \approx J_i A_{ch}/I_d$ is the beam utilization efficiency, or the ratio of ion current to total discharge current. Given appropriate data, we integrate Eqs. 28–30 upstream from the right simulation boundary using a second-order predictor-corrector method. In doing so, we use the same ionization rate coefficients as in *HallThruster.jl*. We then use the results to evaluate the inverse anomalous Hall parameter (Eq. 27).

Lacking the necessary experimental data to apply this method to the SPT-100, we compare it *in-silico* to the diffusion model, using the same reference simulation as in Secs 3.1 and 3.2. As our 1-D fluid simulations do not produce IVDFs to which the Perez-Luna technique can be applied, we assume that E is known in addition to n_e, T_e, and $u_{i,1}$. This ensures parity between our methods. To more faithfully mimic experimental data, we observe these quantities at just 15 data axial locations, chosen to match those of the LIF data from Ref. [18] used in the previous section. Furthermore, we perturb each of these points with relative Gaussian noise with a standard deviation of 2.5%. We additionally condition the diffusion model on a single observation each of n_n and u_e at the final data point in the domain, calculated from Eqs. 32 and 34. The diffusion model is informed of five of the six scalar parameters: V_d, V_{cc}, $\dot{m}_a$, f_B, and u_n, as these are either readily obtained in experiments or, in the case of u_n, required by Roberts' method. As the wall loss coefficient f_w is unknown, we marginalize our predictions over this value by sampling a random value from its prior distribution (Tab. 1) for each diffusion model sample.

Fig. 7 compares the results of the method of Roberts and Jorns to those of the diffusion model. While Roberts' method successfully inverts many of the field quantities, in all cases the median diffusion model predictions more closely match the reference simulation. Moreover, the diffusion model extrapolates past the domain of the measurements toward the anode at $z = 0$ cm and downstream to $z = 8$ cm. Roberts' method is unable to do so, and in the case of the ion densities strongly departs from the reference simulation at the upstream end of the measurement domain. The diffusion model is

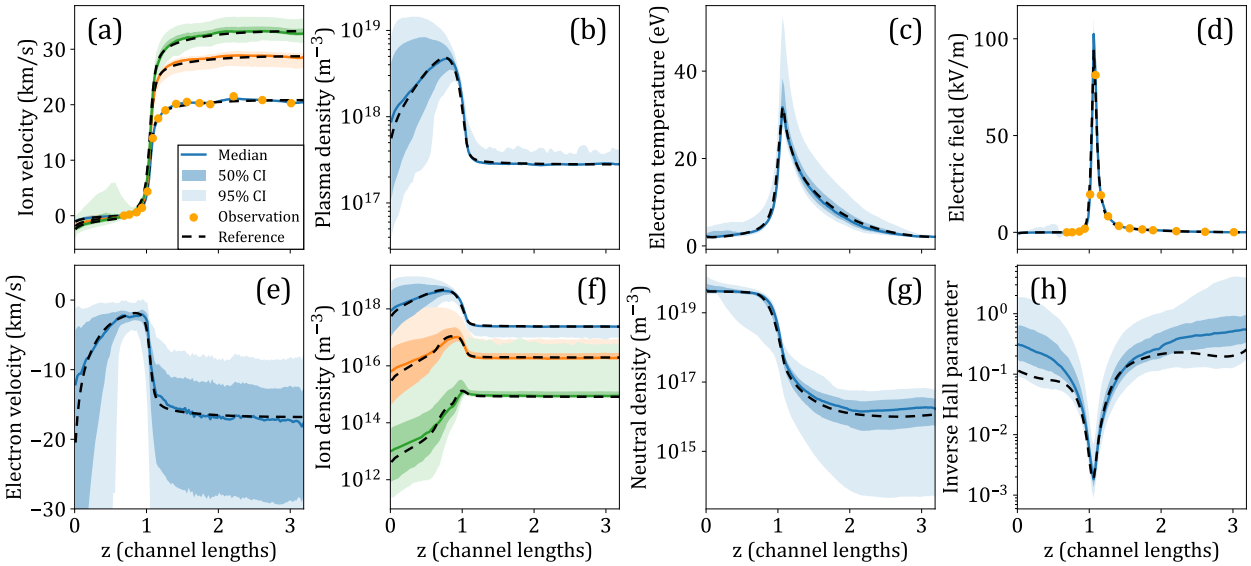


Figure 8: Comparison of the method of Roberts and Jorns the diffusion model has been conditioned only on simulated LIF measurements. Colors are the same as Fig. 7. The uncertainty in the diffusion outputs is higher than when electron density and temperature were included (c.f. Fig. 7), but the median values still match or improve upon those from Roberts' method.

also more tolerant to noise and artifacts caused by numerical differentiation. For example, the inverse Hall parameter recovered by the method of Roberts and Jorns crosses zero twice, at the first and last measurement points. This stems from noise amplification when calculating the pressure gradient term in Eq. 27. The plasma density and electron velocity exhibit the largest uncertainties of the diffusion outputs, likely due to the marginalization of our predictions over f_w. The median electron velocity also closely follows Roberts' prediction and the conditioned value, which differ from the true value due to the approximations made in obtaining the downstream electron velocity using Eq. 32. The electron velocity also exhibits significant noise. This may be a sampling artifact, but may also indicate the presence of transitions between quiescent and large-scale oscillatory modes across the parameter ranges consistent with the data. While our model does not directly capture time dependent effects, simulations that exist near mode transitions sometimes show noisy time-averaged properties due to the instability of the oscillations and short averaging window of $500\,\mu s$.

Despite some of the discrepancies, Roberts' method does an excellent job at field inversion, especially near the middle of the measurement domain. However, this success depends on the availability of ITS measurements of the plasma density and temperature. More commonly, only ion velocity and electric field data from LIF is available. To investigate the effect of missing ITS data, we show in Fig. 8 the results of conditioning the diffusion model solely on the ion velocity and electric field. We observe that while the uncertainty in our estate estimates has increased, as expected, the median values of most values still agree nearly as well with the reference simulation as before. Moreover, while still noisy, the median electron velocity now matches the reference simulation. These results highlight the flexibility of the diffusion model with respect to the availability of data.

4 Discussion

4.1 Inference costs

In this section, we quantify the difference in computational cost (measured in wall time) between the proposed method and traditional Bayesian inference methods like MCMC. The cost of MCMC is straightforward to calculate—to generate N_s independent samples of the posterior, the cost is

$$\text{cost(MCMC)} = N_s \, t_{sample}/\eta_{sample}, \tag{35}$$

where t_{sample} is the time required to propose, evaluate, and accept or reject a sample; and η_{sample} is the sampling efficiency, or the effective sample size divided by the overall number of samples. In our case, t_{sample} is at least the cost required to run a Hall thruster simulation. The use of the DRAM algorithm increases t_{sample}, as delayed rejection leads

to multiple posterior evaluations, and thus simulations, on some samples. Finally, the sampling efficiency accounts for the fact that successive samples are correlated, causing the number of truly independent samples to be far less than the nominal value.

We break the cost of the diffusion model into three phases—data generation, training, and sampling. Data generation can happen in parallel, so the cost of generating N_{train} training examples in parallel batches of B_{data} at a cost of t_{sim} per simulation is

$$\text{cost(generation)} = N_{train}\ t_{sim}/B_{data}. \tag{36}$$

Training also proceeds in batches of size B for N_{epochs} passes through the training data. If t_{batch} is the time to step through one batch of data, then the training cost is equal to

$$\text{cost(training)} = N_{epochs} N_{data} t_{batch}/B. \tag{37}$$

Finally, we sample the diffusion model. Each sample is independent and generated in parallel, but in practice the cost grows linearly after a certain batch size. Assuming purely serial sample generation of S steps and a per-step cost for one sample of t_{step}, the cost of N_S samples is

$$\text{cost(sampling)} = N_S S\ t_{step}. \tag{38}$$

We next turn to estimating these parameters. For MCMC, we take $t_{sample} = 4.5$, assuming a base cost of $t_{sim} = 3$ seconds and that half of the proposed samples trigger delayed rejection. The sampling efficiency can be estimated by dividing the total number of samples by the lag required for autocorrelation to drop to near-zero. We find this to be $O(100)$ for our problem, depending on the parameter investigated. This gives $\eta_{sample} \approx 0.01$. Generating 256 independent samples using MCMC would thus take about 32 hours, neglecting any burn-in phase.

In Appendix A, we find that about 2^{19} training examples are needed for sampling quality to converge. In the present work, we generated these in batches of 24. At $t_{sim} \approx 3$ seconds, data generation would take about 18 hours. Next, model training time depends strongly on the available hardware; using an H100 GPU, it took about 46 hours, while on an RTX 3060 it took 104 hours. Finally, sampling time is comparatively negligible; for a batch of 256 samples using 2048 steps it took 2 minutes, 3 seconds on the H100 and 11 minutes, 54 seconds on the RTX 3060. This gives a total inference cost of 64 hours for the H100 and 122 hours for the RTX 3060. In practice, the data generation cost can be reduced linearly by increasing B_{data}, leaving the model training time as the main irreducible component.

While the costs cited here are larger than MCMC, we have up to this point been discussing the solution of a single inverse problem with a fixed number of samples. Solving a second inverse problem on the same physical system, perhaps by modifying the noise model or changing the data, requires us to re-run MCMC from scratch, doubling the overall time required. In contrast, the diffusion model cost only increases by the time needed to generate the samples. Similarly, generating additional samples on the same problem causes the cost of MCMC to increase linearly, while the diffusion model's cost remains essentially fixed. At 512 independent samples, the cost of MCMC matches that of diffusion model trained on the H100, and at 1024 samples it reaches that of the RTX 3060. On the other hand, increasing the sampling efficiency of MCMC while keeping the cost per sample flat would push the calculus in favor of MCMC. For low-dimensional inference problems where the posteriors are close to Gaussian, the sampling efficiency is likely to be high, and MCMC will dominate methods based on diffusion models. For higher-dimensional inverse problems and problems in which Bayesian inference needs to be performed repeatedly in an online setting, diffusion-based inference is a promising technique.

4.2 Use cases

Given the results presented in this work and the cost discussion laid out above, we now turn to discussing some promising use-cases for diffusion-based inference in $E \times B$ plasma sources. The time to generate and train a diffusion model like the one in this work is far less than the time needed to design, construct, and qualify a thruster, or even set up an experimental test campaign. This means that a model could be trained in parallel with the development and set-up of a test. Once trained, the model could be queried as new data comes in, providing plasma state estimates whose uncertainty would help guide and optimize additional data collection. Such a model would in effect act as a digital twin for the thruster, being retrained or fine-tuned periodically as the test article ages and its behavior changes.

The relative speed of a trained diffusion model relative to classical inference methods in an online setting may enable its integration into a control system. In such an application, the model would assimilate sparse data from a running device to obtain and continuously update a plasma state estimate and inform control actions. The speed of this control loop would be bottlenecked by the sampling speed. At present, this implies control times on the order of minutes. Given the long mission durations and slow acceleration of Hall thrusters, this may be acceptable. However, in light of the speed improvements in image generation models in recent years, however [11], we believe that with some

optimization the number of sampling steps required present model could be reduced by a factor of ten or more. The speed of the underlying 1-D code also allows the diffusion model's outputs to be checked for correctness rapidly. Discrepancies between the forward model and diffusion model could then be used to correct proposed control actions and determine when the thruster or controller has entered an off-nominal state.

4.3 Limitations

Our method currently incorporates scalar parameters such as mass flow rate and discharge voltage in a manner analogous to class labels in image generators, baking them in at training time. As a result, they do not participate in diffusion, preventing us from inferring them from data. This limitation became clear in our comparisons to experimental data in Sec. 3.3, where we were forced to marginalize over the unknown parameters. While functional, this gave overly-pessimistic uncertainty estimates. Similarly, our method currently does not consider time-resolved information. While we could naively incorporate this by stacking multiple time-instants into a 3-D tensor, this would dramatically increase the cost of the method. A promising middle ground would be to embed a latent representation of time-dependent information into the model analogously to the noise level and scalar parameters.

As shown in Appendix A, our model currently requires a large amount of training data (at least 2^{19} examples for the current architecture). While this feasible to generate with an optimized 1-D code like *Hallthruster.jl*, it would be impossible with higher-fidelity codes, even in parallel. It is possible that incorporating physics guidance into the code, either during training or sampling could reduce the data requirements [14, 13]. Alternate architectures based on neural operators may reduce training costs and enhance generalization at reduced data sizes [29].

Finally, diffusion-based inference is limited by the distribution of the training data and the capabilities of the underlying model. If the prior distributions of the model inputs are not chosen to be sufficiently wide, the diffusion model may be unable to be successfully conditioned on data that lies outside the chosen bounds. Additionally, the present diffusion model works only for a single thruster and propellant, and would need to be retrained to solve inverse problems on a new system. Furthermore, missing physics and potential discrepancies between model and data must be accounted for in some way. We attempt to resolve these effects with a highly-flexible parameterization of the anomalous collision frequency curve, but other effects like non-classical heat flux [9] would still be hard to detect. Such effects could themselves be parameterized, but it would be impossible to account for all sources of discrepancy between experiment and data without a prohibitively high-fidelity code. Finally, we note that as we trained our diffusion model on a 1-D code, we are unable to condition it on data originating from off of the simulation axis. This reduces its usefulness in important tasks like erosion prediction.

5 Conclusion

In this work, we have demonstrated the viability of diffusion models for inverting sparse measurements to recover full state estimates in a low temperature, crossed-field plasma device. After training a diffusion model on a dataset of 1-D simulations of a Hall effect thruster, we successfully demonstrated its use as a forward surrogate of the underlying simulations. We then applied it to several probabilistic inference problems involving both real and simulated data and found that it matched or exceeded the performance of semi-analytic methods and compared favorably to ground truth MCMC predictions. Our chosen architecture and sampling procedure support arbitrary conditioning at inference time, without re-training the neural network. This makes it an attractive alternative to classical inference methods, particularly in settings such as data assimilation where inference needs to be performed repeatedly.

With these benefits, there are also some challenges that remain to be solved. Future work will focus on reducing the cost of sampling, incorporation of physics information to reduce data requirements, and a more general treatment of scalar and operating parameters. In view of advancements in physics-informed generative modeling in other fields, these challenges are surmountable. We believe that given their clear advantages in accuracy, online computational cost, and flexibility, models like ours will be increasingly valuable tools in low temperature plasma research.

Acknowledgements

This work was supported by the Laboratory Directed Research and Development program at Sandia National Laboratories, a multimission laboratory managed and operated by National Technology and Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525. Computing resources were provided by an Air Force Office of Scientific Research DURIP under Program Manager Dr. Fariba Fahroo and grant number FA9550-23-1-006,

as well as by Advanced Research Computing at the University of Michigan. The code used to generate the results in this work is available online on GitHub at `https://github.com/archermarx/hall_diffusion`

A Appendix: Ablations

In this section, we investigate the effect of several numerical parameters on the performance of our diffusion model. We first begin by describing the methodology by which we assign a single numerical score to the "quality" of a batch of samples from the diffusion model, before using this metric to quantitatively investigate the role of training data size and sampling parameters on sample quality.

The task of evaluating the quality of samples generated from a diffusion model can be framed as a finding a suitable distance metric between probability distributions. With such a metric in hand and a test set obtained from the true distribution, we can ask how "far apart" the distribution of our samples is from that of the test set. In this work, we choose the Maximum Mean Discrepancy (MMD) as our distance metric [32]. Unlike other common distance metrics used to evaluate image models, such as the Fréchet Inception Distance (FID), the MMD is unbiased, makes no assumptions about the shape of the underlying distribution [33].

The MMD is a kernel-based distance metric and requires a kernel function $k(x, y)$ to measure the similarity to two samples. For two distributions $x \sim p(X)$ and $y \sim q(Y)$, from which we have drawn m and n samples, respectively, an unbiased estimator for d^2_{MMD} is [32]

$$d^2_{\text{MMD}} = \frac{1}{m(m-1)} \sum_{i=1}^{m} \sum_{j \neq i}^{n} k(x_i, x_j) + \frac{1}{n(n-1)} \sum_{i=1}^{n} \sum_{j \neq i}^{n} k(y_i, y_j) - \frac{2}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} .k(x_i, y_j) \tag{39}$$

We use the squared exponential radial basis function kernel, $k(x, y) = \exp\left[-\|x - y\|^2 / 2\ell^2\right]$. The kernel width, ℓ, is a free parameter but has a weak effect on the results. We fix it at 10.

In image generation settings, the chosen distance metric is typically not evaluated in pixel space. Instead, the samples are compressed or embedded into a latent space which is more semantically meaningful. Typically, this is done with a separate image embedding model, such as Inception or CLIP [33]. Such a model does not exist for Hall thrusters. To avoid designing, training, and testing one an entirely separate neural network from scratch, we turn to tensor-based compression methods. Specifically, we use the incremental Hierarchical Tucker (HT) algorithm developed by Aksoy [34], to compress our samples. We run the algorithm on our training dataset of 2^{21} examples to generate the HT compression cores. These which can then be used to compress all generated samples, as well a a held-out test set, into the same latent space. We set a reconstruction error tolerance of 10% when compressing the training dataset, which gave a compression factor of 6x.

With these preliminaries developed, our procedure for these ablations is as follows. We first create the HT compressor on the training dataset and use it compress a test set. Next, we generate samples from a diffusion model using the parameters under study for the ablation. Finally, we compute d^2_{MMD} between the generated samples and the test set using $m = n = 4096$ samples of each dataset. For uncertainty quantification, we evaluate d^2_{MMD} 12 times on different sets of 4096 samples and report 50% and 95% credible intervals. We study four parameters — training data size, model size, derivative loss scale, and number of sampling steps, on both unconditional and conditional sample generation. For unconditional sample generation, we compare our samples to a test dataset of 131,072 simulations generated in the same way as in Sec. 2.1 which were not included in the training dataset. Meanwhile, we generate conditional samples using the ion velocity profile from the reference simulation as in Sec. 3.2. These are then compared to the samples obtained using MCMC from the same section. We present the results of these ablations in Fig. A1, and describe the details of each study in the following sections.

Training data size

One of the main challenges with applying any machine learning technique is the need for large amounts of data to train a sufficiently powerful model for the task at hand. While in our case the model was cheap enough that we could generate millions of training examples, generating a similar amount for a 2-D or 3-D physics model would be computationally infeasible. To assess the data needs of our method, we train multiple models on differently-sized subsets of the full training dataset of 2^{21} samples During training, each model saw the same number of total examples, i.e. for smaller dataset sizes, the model made more passes through the training data. In Fig. A1a and Fig. A1e, we report the effect of varying the number of simulations used to train the model on d^2_{MMD}. We find that generation quality improves with increasing data size and saturates at a size of 2^{19} (524,288) training examples for unconditional sampling and 2^{17} (131,072) examples for conditional sampling. The reduced data requirements for conditional sampling in this case likely stem from the fact that our chosen reference simulation is a relatively "typical"

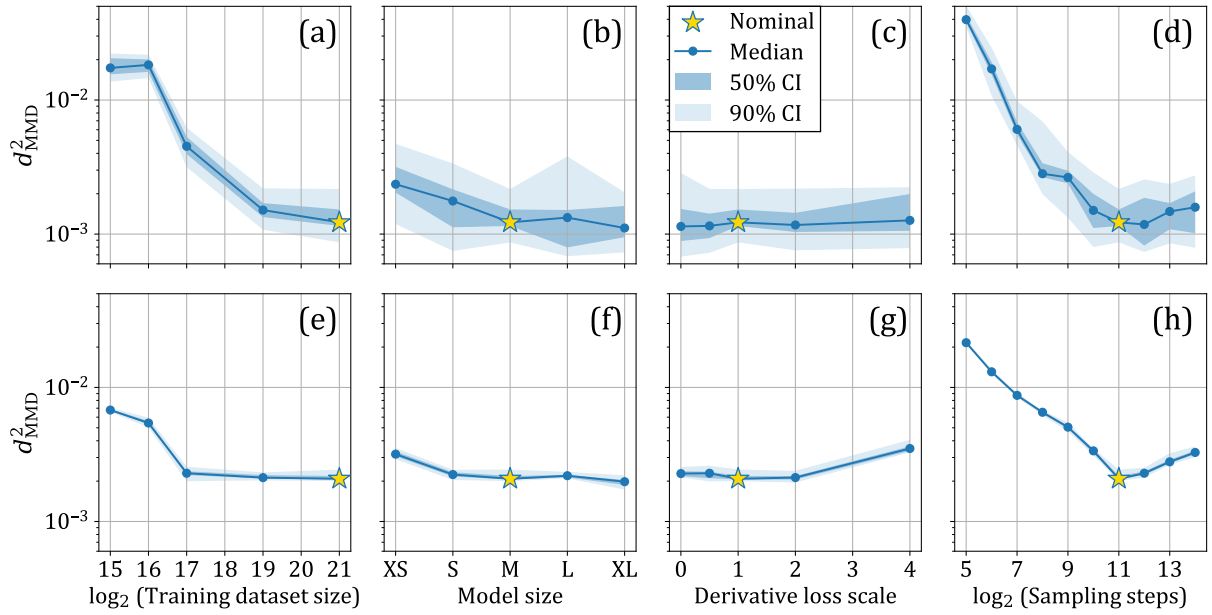


Figure A1: Squared maximum mean discrepancy (d^2_{MMD}) as a function of training data size, model size, derivative loss scale and number of sampling steps. The top row shows results for unconditional sampling using 4096 samples, compared to an equal number from a test dataset. The bottom row corresponds to conditional sampling with a reference of 4096 samples from a Markov Chain Monte Carlo run as in Sec. 3.2. The nominal parameter values used in the body of the paper are indicated with a yellow star. Generation quality is most strongly affected by the amount of training data available and the number of sampling steps taken.

Table A1: Parameters, batch sizes, and training times for model size ablations.

Model	Blocks per layer (N_B)	Encoding channels (N_C)	Batch size	Training time (H100)
XS	2	32	8192	1 day, 16 hours
S	2	64	8192	1 day, 19 hours
M	3	64	8192	1 day, 22 hours
L	4	64	8192	2 days, 6 hours
XL	4	96	4096	7 days, 8 hours

Hall thruster simulation. Conditional sampling on this data thus guides sampling toward higher-probability regions of the prior probability distribution which can be learned with less data. In both cases, the required number of samples is large and only possible to obtain due to the low computational cost of our Hall thruster code. Physics guidance may help reduce the data requirements further by exploiting known relationships between fields during sampling [13]. Additionally, architectures based on neural operators may help improve generalization at lower dataset sizes and reduce data generation and training costs by allowing simulations of multiple resolutions to be used simultaneously [29].

Model size

We next investigate the role of model size on the sample quality. For this, we define five models: **XS**, **S**, **M**, **L**, **XL**, with parameters given in Tab. A1, and train them on the full dataset of 2^{21} samples. The **M** model corresponds to the model used throughout the body of the paper. We vary the number of residual blocks per layer (N_B) as well as the number of encoding channels (N_C). Each model was trained on the same number of total training examples at the maximum batch size that could fit in the memory of a single Nvidia H100 GPU [35]. We plot in Fig. A1b and Fig. A1f the effect of model size on d^2_{MMD}. Increasing model size leads to improved performance for both conditional and unconditional sampling, but the effect is subtle and saturates past the **M** case. The median discrepancy of the **XS** model is 2 times that of the **M** model, while the **XL** model offers no further improvement outside of uncertainty. This suggests that the features of our dataset are more or less optimally captured by the **M** model.

Derivative terms in loss function

In contrast to prior diffusion modeling efforts, we incorporate derivative terms into our loss function. The intention of this is to ensure smooth gradients and to reduce point-to-point noise in generated samples. Figures A1c and A1g show the results of varying the derivative factor in the loss term, f_d, from 0 to 4. We find that $f_d \leq 2$ does not impact generation quality, while $f_d = 4$ harms conditional sample generation. While the effect of these derivative terms in the present study was marginal, they may still provde useful when applying physics guidance in future work.

Sampling steps

One of the drawbacks of diffusion models compared to other image generation methods is the relatively long sampling time. This stems from the iterative time-stepping approach used for sampling compared to the one-step method used by architectures like variational auto-encoders (VAEs) and generative adversarial networks (GANs). Reducing the number of sampling steps taken directly reduces the cost of the method, at the cost of worse sample generation and in some cases reduced sampling stability. Figures A1d and A1h plot the sample quality versus number of sampling steps taken. We find that d^2_{MMD} decreases with increasing sampling steps up to 2^{11} (2048), which was which was the nominal value used in this work. Above this value, the sampling quality degrades to the same level as at 1024 sampling steps. As the sampling process is sequential, the time to generate samples increases approximately linearly with the number of sampling steps, with the model taking 42 seconds to generate 1024 samples at 256 steps and 2040 seconds at 16384 steps.

References

[1] Igor D. Kaganovich, Andrei Smolyakov, Yevgeny Raitses, Eduardo Ahedo, Ioannis G. Mikellides, Benjamin Jorns, Francesco Taccogna, Renaud Gueroult, Sedina Tsikata, Anne Bourdon, Jean-Pierre Boeuf, Michael Keidar, Andrew Tasman Powis, Mario Merino, Mark Cappelli, Kentaro Hara, Johan A. Carlsson, Nathaniel J. Fisch, Pascal Chabert, Irina Schweigert, Trevor Lafleur, Konstantin Matyash, Alexander V. Khrabrov, Rod W. Boswell, and Amnon Fruchtman. Physics of E×B discharges relevant to plasma propulsion and similar technologies. *Physics of Plasmas*, 27(12):120601, 12 2020.

[2] Jean-Pierre Boeuf. Tutorial: Physics and modeling of Hall thrusters. *Journal of Applied Physics*, 121(1):011101, 01 2017.

[3] Ioannis G. Mikellides and Alejandro Lopez Ortega. Challenges in the development and verification of first-principles models in Hall-effect thruster simulations that are based on anomalous resistivity and generalized Ohm's law. *Plasma Sources Science and Technology*, 28:48, 1 2019.

[4] D. E. Troyetsky, A. Lopez Ortega, and K. Hara. Gradient-based calibration of the anomalous electron scattering frequency in a Hall effect thruster simulation. *Journal of Applied Physics*, 138(16):163302, October 2025.

[5] Joshua D. Eckels, Thomas A. Marks, Madison G. Allen, Benjamin A. Jorns, and Alex A. Gorodetsky. Hall thruster model improvement by multidisciplinary uncertainty quantification. *Journal of Electric Propulsion*, 3(19), 2024.

[6] J Pérez-Luna, G J M Hagelaar, L Garrigues, and J P Boeuf. Method to obtain the electric field and the ionization frequency from laser induced fluorescence measurements. *Plasma Sources Science and Technology*, 18(3):034008, jul 2009.

[7] Ethan Dale. *Investigation of the Hall Thruster Breathing Mode*. PhD thesis, University of Michigan, 2019.

[8] Ethan T. Dale and Benjamin A. Jorns. Non-invasive time-resolved measurements of anomalous collision frequency in a Hall thruster. *Physics of Plasmas*, 26, 2019.

[9] Parker J. Roberts and Benjamin A. Jorns. Laser measurement of anomalous electron diffusion in a crossed-field plasma. *Phys. Rev. Lett.*, 132:135301, Mar 2024.

[10] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Proc. NeurIPS*, 2020.

[11] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Proc. NeurIPS*, 2022.

[12] Giannis Daras, Hyungjin Chung, Chieh-Hsin Lai, Yuki Mitsufuji, Jong Chul Ye, Peyman Milanfar, Alexandros G. Dimakis, and Mauricio Delbracio. A survey on diffusion models for inverse problems, 2024.

[13] Jiahe Huang, Guandao Yang, Zichen Wang, and Jeong Joon Park. DiffusionPDE: Generative PDE-solving under partial observation. *Advances in Neural Information Processing Systems*, 37:130291–130323, 2024.

[14] Christian Jacobsen, Yilin Zhuang, and Karthik Duraisamy. Cocogen: Physically consistent and conditioned score-based generative models for forward and inverse problems. *SIAM Journal on Scientific Computing*, 47(2):C399–C425, 2025.

[15] Thomas Marks, Paul Schedler, and Benjamin Jorns. HallThruster.jl: a Julia package for 1-D Hall thruster discharge simulation. *Journal of Open Source Software*, 8(86):4672, 2023.

[16] John M Sankovic, John A Hamley, Thomas W Haaa, and Thomas W Haagt. Performance evaluation of the Russian SPT-100 thruster at NASA LeRC. In *1993 International Electric Propulsion Conference*, page 094, 1993.

[17] Jean Pierre Boeuf and Andrei Smolyakov. LANDMARK plasma test cases. `https://jpb911.wixsite.com/landmark/test-cases`, 2019. Accessed: 2024-12-02.

[18] Natalia MacDonald-Tenenbaum, Quinn Pratt, Michael Nakles, Nickolas Pilgram, Michael Holmes, and William Hargus. Background pressure effects on ion velocity distributions in an SPT-100 Hall thruster. *Journal of Propulsion and Power*, 35:403–412, 2019.

[19] Thomas A. Marks, Joshua D. Eckels, Gabriel A. Mora, and Alex A. Gorodetsky. Uncertainty quantification of a multi-component Hall thruster model at varying facility pressures. *Journal of Applied Physics*, 138(15), 2025.

[20] Justin W. Koo and Iain D. Boyd. Modeling of anomalous electron mobility in Hall thrusters. *Physics of Plasmas*, 13:3541, 3 2006.

[21] Richard Hofer, Ira Katz, Dan Goebel, Kristina Jameson, Regina Sullivan, Lee Johnson, and Ioannis Mikellides. Efficacy of electron mobility models in hybrid-PIC Hall thruster simulations. In *44th AIAA/ASME/SAE/ASEE Joint Propulsion Conference*. AIAA, 2012.

[22] J. Perales-Díaz, A. Domínguez-Vázquez, P. Fajardo, E. Ahedo, F. Faraji, M. Reza, and T. Andreussi. Hybrid plasma simulations of a magnetically shielded Hall thruster. *Journal of Applied Physics*, 131(10):103302, 03 2022.

[23] Ioannis G. Mikellides and Ira Katz. Numerical simulations of Hall-effect plasma accelerators on a magnetic-field-aligned mesh. *Physical Review E*, 86, 10 2012.

[24] Thomas A. Marks and Benjamin A. Jorns. Challenges with the self-consistent implementation of closure models for anomalous electron transport in fluid simulations of Hall thrusters. *Plasma Sources Sci. Technol.*, 32(4):045016, 4 2023.

[25] M. Loeve. *Probability Theory II*. F.w.Gehring p.r.Halmos and c.c.Moore. Springer, 1978.

[26] Tero Karras, Miika Aittala, Jaakko Lehtinen, Janne Hellsten, Timo Aila, and Samuli Laine. Analyzing and improving the training dynamics of diffusion models. In *Proc. CVPR*, 2024.

[27] Jiaming Song, Arash Vahdat, Morteza Mardani, and Jan Kautz. Pseudoinverse-guided diffusion models for inverse problems. In *International Conference on Learning Representations*, September 2022.

[28] Hyungjin Chung, Jeongsol Kim, Michael T. Mccann, Marc L. Klasky, and Jong Chul Ye. Diffusion posterior sampling for general noisy inverse problems, 2024.

[29] Jiachen Yao, Abbas Mammadov, Julius Berner, Gavin Kerrigan, Jong Chul Ye, Kamyar Azizzadenesheli, and Anima Anandkumar. Guided diffusion sampling on function spaces with applications to PDEs, November 2025.

[30] Heikki Haario, Marko Laine, Antonietta Mira, and Eero Saksman. DRAM: Efficient adaptive MCMC. *Statistics and Computing*, 16:339–354, 12 2006.

[31] Kevin D. Diamant, Raymond Liang, and Ron L. Corey. The effect of background pressure on SPT-100 Hall thruster performance. In *50th AIAA/ASME/SAE/ASEE Joint Propulsion Conference*, AIAA Propulsion and Energy Forum. American Institute of Aeronautics and Astronautics, 2014. AIAA 2014-3710.

[32] Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *J. Mach. Learn. Res.*, 13(null):723–773, March 2012.

[33] Sadeep Jayasumana, Srikumar Ramalingam, Andreas Veit, Daniel Glasner, Ayan Chakrabarti, and Sanjiv Kumar. Rethinking FID: Towards a Better Evaluation Metric for Image Generation, January 2024.

[34] Doruk Aksoy and Alex A. Gorodetsky. Incremental Hierarchical Tucker Decomposition, December 2024.

[35] Nvidia Corporation. *NVIDIA H100 Tensor Core GPU Datasheet*, 2024. Accessed: 2024-06-18.