

A Proof of Hard Constraint Satisfaction

Appendix A: Proof of Hard Constraint Satisfaction

We now show that, under standard regularity assumptions, PCFM (see Algorithm 1) returns a final sample u_1 satisfying the hard constraint $h(u_1) = 0$ to numerical precision.

Theorem A.1 (Exact Constraint Enforcement). *Let $h : \mathbb{R}^n \rightarrow \mathbb{R}^m$ be a twice continuously differentiable constraint function, and suppose the Jacobian $J_h(u) := \nabla h(u) \in \mathbb{R}^{m \times n}$ has full row rank $m \leq n$ in a neighborhood of the constraint manifold $\mathcal{M} := \{u \in \mathbb{R}^n : h(u) = 0\}$. Then the final sample u_1 produced by PCFM satisfies*

$$h(u_1) = 0$$

to machine precision, provided the final projection step in Algorithm 1 is solved using sufficiently many Newton–Schur iterations.

Proof. We consider the final correction step in Algorithm 1:

$$u_1 := \arg \min_{u'} \|u' - u_1\|^2 \quad \text{s.t.} \quad h(u') = 0,$$

which seeks to project the current sample u_1 (obtained after flow integration and relaxed penalty correction) onto the feasible manifold.

Let $u^{(0)} := u_1$ denote the initial guess to the Newton solver. The constrained root-finding problem $h(u) = 0$ is nonlinear, but under the full-rank Jacobian assumption, the *implicit function theorem* ensures a locally unique root $u^* \in \mathcal{M}$ near $u^{(0)}$. We solve the nonlinear system using the following Newton–Schur iteration:

$$u^{(k+1)} = u^{(k)} - J_h(u^{(k)})^\top \left[J_h(u^{(k)}) J_h(u^{(k)})^\top \right]^{-1} h(u^{(k)}). \quad (\text{A.1})$$

This corresponds to the batched Schur-complement Newton update implemented in our differentiable solver. Standard convergence theory (e.g., Theorem 10.1 in [66]) guarantees that, for sufficiently small $\|h(u^{(0)})\|$, the iterates $u^{(k)}$ converge *quadratically* to u^* . Hence, for any tolerance $\varepsilon > 0$, there exists a finite K such that

$$\|h(u^{(K)})\| < \varepsilon.$$

Since we continue iterations until $\|h(u^{(K)})\| < \varepsilon_{\text{tol}}$ (with ε_{tol} typically set to machine precision), we conclude

$$h(u_1) = h(u^{(K)}) \approx 0.$$

Finally, we verify that the initial guess $u^{(0)} = u_1$ is within the *basin of convergence*. Each prior step of PCFM includes: (i) a Gauss–Newton projection, (ii) an OT-based backward solve, and (iii) a relaxed penalty correction. These ensure the incoming residual $\|h(u_1)\|$ is already small (typically $< 10^{-3}$). Combined with the well-conditioned Jacobian J_h , we ensure convergence of the final Newton–Schur projection. □

B PDE Dataset Details

We evaluate our method on four representative PDE systems commonly studied in scientific machine learning: the **Heat Equation**, **Navier-Stokes Equation**, **Reaction-Diffusion Equation**, and **Burgers' Equation**. These problems cover linear and nonlinear dynamics, have smooth or discontinuous solutions, and include a variety of initial and boundary condition types.

B.1 Heat Equation

We consider the one-dimensional heat equation with periodic boundary conditions, following the setting in Hansen et al. [13]:

$$u_t = \alpha u_{xx}, \quad x \in [0, 2\pi], \quad t \in [0, 1], \quad (8)$$

892 with the sinusoidal initial condition

$$u(x, 0) = \sin(x + \varphi), \quad (9)$$

893 and periodic boundary conditions $u(0, t) = u(2\pi, t)$. The analytical solution is given by $u(x, t) =$
 894 $\exp(-\alpha t) \sin(x + \varphi)$. To pretrain our FFM model, we construct the dataset by varying the diffusion
 895 coefficient $\alpha \sim \mathcal{U}(1, 5)$ and phase $\varphi \sim \mathcal{U}(0, \pi)$. All solutions are on a 100×100 spatiotemporal
 896 grid. During constrained sampling, we restrict to the same initial condition by fixing $\varphi = \pi/4$. The
 897 global mass conservation constraint is enforced via

$$\int_0^{2\pi} u(x, t) dx = 0, \quad \forall t \in [0, 1]. \quad (10)$$

898 B.2 Navier-Stokes Equation

899 We consider the 2D incompressible Navier–Stokes (NS) equation in vorticity form with periodic
 900 boundary conditions, following Li et al. [22]:

$$\partial_t w(x, t) + u(x, t) \cdot \nabla w(x, t) = \nu \Delta w(x, t) + f(x), \quad (11)$$

$$\nabla \cdot u(x, t) = 0, \quad (12)$$

901 where $w = \nabla \times u$ denotes the vorticity and $\nu = 10^{-3}$ is the viscosity. The initial vorticity
 902 w_0 is sampled from a Gaussian random field, and the forcing function is defined as: $f(x) =$
 903 $0.1\sqrt{2} \sin(2\pi(x_1 + x_2) + \phi)$, $\phi \sim \mathcal{U}(0, \pi/2)$. We solve the NS system using a Crank–Nicolson
 904 spectral solver on a 64×64 periodic grid, recording 50 uniformly spaced temporal snapshots over
 905 the interval $T = 49$. For training, we generate 10000 simulations by sampling 100 random initial
 906 vorticities and 100 forcing phases. For test dataset, we sample an additional 10 vorticities and 100
 907 forces, yielding 1000 solutions. We fix to 1 initial vorticity for comparison during sampling.

908 Here, we prove the global mass conservation for this NS problem set-up. Under periodic boundary
 909 conditions, global vorticity is conserved. Specifically, integrating the vorticity equation over the
 910 spatial domain Ω and applying the divergence theorem yields:

$$\frac{d}{dt} \int_{\Omega} w(x, t) dx = \int_{\Omega} \nu \Delta w(x, t) + f(x) - u \cdot \nabla w(x, t) dx = 0,$$

911 since the divergence and Laplacian terms vanish due to periodicity and the incompressibility condition
 912 $\nabla \cdot u = 0$. Thus, the total vorticity $\int_{\Omega} w(x, t) dx$ remains constant for all t .

913 B.3 Reaction-Diffusion Equation

914 We consider a nonlinear 1D reaction-diffusion equation with Neumann boundary conditions:

$$u_t = \rho u(1 - u) - \nu \partial_x u, \quad (13)$$

915 on the domain $x \in [0, 1], t \in [0, 1]$, where $(\rho, \nu) = (0.01, 0.005)$. The boundary fluxes at the left
 916 and right ends are specified as g_L and g_R , respectively. The initial condition $u(x, 0)$ is sampled from
 917 a randomized combination of sinusoidal and localized bump functions (see Appendix D for full
 918 specification). The training dataset is constructed by pairing 80 initial conditions with 80 boundary
 919 conditions, producing 6400 PDE solutions, all discretized on a $(nx, nt) = (128, 100)$ space-time
 920 grid. We use a semi-implicit finite difference solver with CFL-controlled time stepping. During
 921 constrained sampling, we fix one initial condition and vary the boundary fluxes to enforce constraint
 922 satisfaction.

923 As this system involves nonlinear source and flux terms, global conservation is no longer trivial.
 924 Integrating both sides of the PDE over the domain yields:

$$\frac{d}{dt} \int_0^1 u(x, t) dx = \rho \int_0^1 u(1 - u) dx + g_L(t) - g_R(t), \quad (14)$$

925 which depends nonlinearly on the state $u(x, t)$ and boundary fluxes g_L, g_R . Thus, conservation must
 926 be tracked through both the nonlinear reaction term and boundary-driven transport, making hard
 927 constraint enforcement via our framework necessary.

928 B.4 Inviscid Burgers' Equation

929 We consider the 1D inviscid Burgers' equation with mixed Dirichlet and Neumann boundary condi-
930 tions:

$$u_t + \frac{1}{2}(u^2)_x = 0, \quad x \in [0, 1], \quad t \in [0, 1], \quad (15)$$

931 with initial condition defined as a smoothed step function centered at a random location $p_{\text{loc}} \sim$
932 $\mathcal{U}(0.2, 0.8)$:

$$u(x, 0) = \frac{1}{1 + \exp\left(\frac{x - p_{\text{loc}}}{\epsilon}\right)}, \quad \epsilon = 0.02. \quad (16)$$

933 We apply a Dirichlet condition on the left, $u(0, t) = u_{\text{bc}}$ with $u_{\text{bc}} \sim \mathcal{U}(0, 1)$, and a Neumann
934 condition (zero-gradient) on the right. Solutions are computed using a Godunov finite volume method
935 on a $(nx, nt) = (101, 101)$ grid. For pretraining the FFM model, we vary both initial and boundary
936 conditions, generating 80 variants each to create 6400 solutions in the training set. For constrained
937 sampling, we use two configurations: (1) fixing the initial condition and varying the boundary
938 condition, and (2) fixing the boundary condition and varying the initial condition.

939 The conservation constraint is nonlinear and particularly sensitive due to shock formation. Integrating
940 the PDE yields

$$\frac{d}{dt} \int_0^1 u(x, t) dx = - \left[\frac{1}{2} u(1, t)^2 - \frac{1}{2} u(0, t)^2 \right],$$

941 so global conservation requires $u(1, t)^2 = u(0, t)^2$, making the constraint highly sensitive to
942 boundary-induced discontinuities.

943 C FFM Model Pretraining

944 For all PDE benchmarks, we employ the functional flow matching (FFM) [10] framework and
945 parameterize the underlying time-dependent vector field with a Fourier neural operator (FNO)
946 backbone [22]. The FNO encoder takes the input the concatenation of the current state u_τ , a sinusoidal
947 positional embedding for the spatial grid and Fourier time embedding. For each benchmark, we adopt
948 the following training scheme unless otherwise specified:

- 949 • Optimizer: Adam optimizer with learning rate 3×10^{-4} , $\beta_1 = 0.9$, $\beta_2 = 0.999$, and no
950 weight decay.
- 951 • Learning rate scheduler: Reduce-on-plateau scheduler with a factor of 0.5, patience of 10
952 validation steps, and a minimum learning rate of 1×10^{-4} .
- 953 • Batch size: 256 for 1D problems (Heat, Reaction-Diffusion, Burgers) and 24 for the 2D
954 Navier-Stokes problem.

955 For Heat, we follow Cheng et al. [11] to train on 5000 analytical solutions of the heat equation. For
956 Reaction-Diffusion and Burgers, we train each model on 6400 numerical PDE solutions. For these 3
957 problems, the FNO is configured with 4 Fourier layers with 32 Fourier modes for both spatial and
958 temporal dimensions, 64 hidden channels, and 256 projection channels. For Navier-Stokes equation,
959 we train a 3D FFM on 10,000 numerical solutions. The FNO has 2 Fourier layers with 16 Fourier
960 modes per dimension, 32 hidden channels, and 256 projection channels, following Cheng et al. [11].
961 FFM models for Heat, Reaction-Diffusion, and Burgers are each trained over 20,000 steps on 1
962 NVIDIA V100 GPU and the Navier-Stokes model is trained for 500,000 steps on 4 NVIDIA A100
963 GPUs.

964 Following Kerrigan et al. [10], all FFM models use a Gaussian prior noise $\mathcal{P}(u_0)$. For all 1D
965 problems, we adopt a *Matern Gaussian Process* prior with smoothness parameter $\nu = 0.5$, kernel
966 length 0.001, and variance 1.0, implemented using GPyTorch. For Navier-Stokes (2D), we adopt
967 standard Gaussian white noise as the prior, $\mathcal{P}(u_0) = \mathcal{N}(0, I)$. Importantly, PCFM is applied post-hoc
968 at inference time, allowing us to reuse the pretrained unconditional FFM models across constraint
969 configurations.

D Constraints Enforcement via PCFM

We focus on physical and geometric constraints of the form $\mathcal{H}u = 0$, which must be satisfied as equality conditions on the solution field u . First, we describe common equality constraints encountered in solving PDEs. Next, we describe specific constraint setup for our PDE datasets.

D.1 Linear Equality Constraints

Dirichlet Initial and Boundary Conditions Constraints such as $u(x, 0) = \alpha_0(x)$ or $u(x, t) = g(x, t)$ on $\partial\Omega \times [0, T]$ define fixed-value conditions that are linear in u . These can be encoded in the general form

$$\mathcal{H}u = Au - b = 0, \quad (17)$$

where A is a sampling or interpolation operator applied to u , and b is the prescribed target value.

Linear Global Conservation Laws (Periodic Systems) In systems with periodic boundary conditions, additive invariants such as total mass are often linearly conserved. A representative example is

$$\mathcal{H}u = \int_{\Omega} u(x, t) dx - C = 0, \quad (18)$$

where $C \in \mathbb{R}$ is the conserved quantity. Such constraints can be exactly enforced via analytically derived projections [13], geometric integrators [56], and numerical optimization [65, 47].

D.2 Nonlinear Equality Constraints

Many physical systems exhibit global conservation laws where the conserved quantity depends nonlinearly on u . A representative form is:

$$\mathcal{H}u = \frac{d}{dt} \int_{\Omega} \rho(u(x, t)) dx = 0, \quad (19)$$

where $\rho(u)$ denotes a nonlinear density, such as total energy or entropy. While such constraints are often not conserved exactly by standard numerical integrators, structure-preserving methods, such as symplectic or variational integrators [56], can conserve invariants approximately over long time horizons. In our setting, however, we seek to enforce such nonlinear conservation laws exactly at inference time and moreover only at the final denoised sample. Thus, we rely on applying iterative projection methods such as Newton’s method for constrained least squares [65] to satisfy Equation (19) up to numerical tolerance.

Neumann-Type Boundary Constraints Boundary constraints involving fluxes, such as

$$\mathcal{H}u = \partial_n u(x, t) - g(x, t) = 0, \quad (20)$$

are typically linear under standard semi-discretizations, as the derivative operator is linear. However, these constraints can become effectively nonlinear in practice when implemented with upwinding or other nonlinear flux-discretization schemes, particularly in hyperbolic PDEs [57]. Additionally, if the prescribed flux g depends nonlinearly on u , the constraint becomes explicitly nonlinear. In both cases, we handle them using differentiable correction procedures or projection-based updates within our framework.

In our PCFM framework, the residual operator $\mathcal{H}(u)$ specifies the equality constraints to be enforced on the generated solution u . We enforce these constraints on the generated solution state in our PCFM projection operator, ensuring that the final solution strictly satisfies the required physical constraints without retraining the underlying flow model. We summarize the specific constraint formulations used for each PDE task below.

D.3 Heat Equation (IC and Mass Conservation)

The residual is a concatenation of two linear constraints:

$$\mathcal{H}_{\text{Heat}}(u) = \left[\begin{array}{c} u(x, t=0) - u_{\text{IC}}(x) \\ \int u(x, t) dx - \int u(x, t=0) dx \end{array} \right] \quad (21)$$

where the first term enforces the initial condition and the second term ensures mass conservation over time.

1010 **D.4 Navier-Stokes Equation (IC and Mass Conservation)**

1011 For the 2D Navier-Stokes problem, we impose the initial vorticity and the global mass conservation
1012 constraint over the spatial domain:

$$\mathcal{H}_{\text{NS}}(u) = \left[\iint u(x, y, t) dx dy - \iint u(x, y, t = 0) dx dy \right] \quad (22)$$

1013 where $u(x, y, t)$ represents the vorticity field over spatial coordinates (x, y) and time t . The mass
1014 conservation term enforces that the total vorticity integrated over the spatial domain remains consistent
1015 with the initial condition over all time steps.

1016 **D.5 Reaction-Diffusion (IC and Nonlinear Mass Conservation)**

1017 We enforce both the initial condition and nonlinear mass conservation which accounts for reaction
1018 and boundary flux terms:

$$\mathcal{H}_{\text{RD}}(u) = \left[m(t) - \left(m(0) + \int_0^t \rho(u) dt + \int_0^t (g_L - g_R) dt \right) \right] \quad (23)$$

1019 where $m(t) = \int u(x, t) dx$ is the total mass, $\rho(u) = \rho u(1 - u)$ is the nonlinear reaction source term,
1020 and g_L, g_R are the Neumann boundary fluxes.

1021 **D.6 Burgers (BC and Mass Conservation)**

1022 We enforce both the boundary conditions (Dirichlet and Neumann) and nonlinear mass conservation:

$$\mathcal{H}_{\text{Burgers-BC}}(u) = \left[\begin{array}{c} u(x = 0, t) - u_L \\ u(x = -1, t) - u(x = -2, t) \\ \int u(x, t) dx - \int u(x, t = 0) dx \end{array} \right] \quad (24)$$

1023 where u_L is the Dirichlet boundary value at the left boundary, and the Neumann BC enforces a
1024 zero-gradient at the right boundary.

1025 **D.7 Burgers (IC, Mass Conservation, and Local Flux)**

1026 We impose three complementary constraints for the Burgers equation: (i) the initial condition, (ii)
1027 global nonlinear mass conservation, and (iii) a sequence of local conservation updates based on
1028 Godunov's flux method. Specifically, the residual is formulated as:

$$\mathcal{H}_{\text{Burgers-IC}}(u) = \left[\begin{array}{c} u(x, t = 0) - u_{\text{IC}}(x) \\ \mathcal{R}_{\text{Flux}}^{(k)}(u) \\ \int u(x, t) dx - \int u(x, t = 0) dx \end{array} \right] \quad (25)$$

1029 where $\mathcal{R}_{\text{Flux}}^{(k)}(u)$ applies k unrolled discrete updates (collocation points) based on Godunov flux:

$$\mathcal{F}(u_L, u_R) = \begin{cases} \min\left(\frac{1}{2}u_L^2, \frac{1}{2}u_R^2\right) & \text{if } u_L \leq u_R \\ \frac{1}{2}u_L^2 & \text{if } u_L > u_R \text{ and } \frac{u_L + u_R}{2} > 0 \\ \frac{1}{2}u_R^2 & \text{if } u_L > u_R \text{ and } \frac{u_L + u_R}{2} \leq 0 \end{cases} \quad (26)$$

1030 We apply $k \in \{1, \dots, 5\}$ unrolled steps to incrementally enforce local conservation dynamics
1031 alongside global mass conservation and initial condition satisfaction.

1032 **E Details of the PCFM**

1033 We provide additional theoretical context for the PCFM algorithm. In Proposition [E.1](#), we formalize
1034 the projection step as a tangent-space update in Hilbert spaces and show that it corresponds to an
1035 orthogonal projection onto the linearized constraint manifold, justifying its use for enforcing both
1036 linear and nonlinear constraints.

1037 **Proposition E.1** (Tangent-Space Projection in Hilbert Spaces). *Let \mathcal{H} be a real Hilbert space and let*
 1038 *$h : \mathcal{H} \rightarrow \mathbb{R}^m$ be a Fréchet-differentiable constraint operator. Consider a point $u_1 \in \mathcal{H}$, and define*
 1039 *the Jacobian $J := Dh(u_1) \in \mathcal{L}(\mathcal{H}, \mathbb{R}^m)$, the bounded linear operator representing the Fréchet*
 1040 *derivative.*

1041 *Then, the update*

$$u_{\text{proj}} = u_1 - J^*(JJ^*)^{-1}h(u_1)$$

1042 *is the unique solution to the constrained minimization problem*

$$\min_{u \in \mathcal{H}} \|u - u_1\|_{\mathcal{H}}^2 \quad \text{subject to} \quad h(u_1) + J(u - u_1) = 0,$$

1043 *and corresponds to the orthogonal projection of u_1 onto the affine subspace defined by the lineariza-*
 1044 *tion of h at u_1 , i.e., the tangent space to the constraint manifold at u_1 .*

1045 *In the special case where $h(u) = Au - b$ is affine, with $A \in \mathcal{L}(\mathcal{H}, \mathbb{R}^m)$, the update becomes the*
 1046 *exact projection onto the feasible set $\{u \in \mathcal{H} : Au = b\}$.*

1047 F Connection to ECI as a Special Case

1048 The ECI (Extrapolate–Correct–Interpolate) framework [11] only considers and showcases empirical
 1049 results on constraints such as fixed initial conditions, Dirichlet boundary values, and global mass
 1050 integrals. These are all linear and non-overlapping, and are applied at isolated subsets of the solution
 1051 field. To be used with the ECI framework, such constraints must admit closed-form oblique projections
 1052 and act independently across disjoint regions. Let us denote any value or regional constraint [11] as a
 1053 linear constraint $h(u) = Au - b$, for some matrix $A \in \mathbb{R}^{m \times n}$, vector $b \in \mathbb{R}^m$, and for simplicity and
 1054 comparison purposes, the ECI hyperparameters such as re-sampling is set to its default value (0) and
 1055 mixing to be 1.

1056 In this regime, our Gauss–Newton update during PCFM guidance simplifies to an exact constraint
 1057 satisfaction step for linear constraints:

$$u_{\text{proj}} = u_1 - A^\top(AA^\top)^{-1}(Au_1 - b),$$

1058 which is precisely the ”correction” step ECI employs. Moreover, by disabling the relaxed penalty
 1059 ($\lambda = 0$) and skipping intermediate corrections, PCFM reduces exactly to ECI.

1060 Hence, ECI is a special case of PCFM, tailored to simple linear constraints, while PCFM provides
 1061 a general and principled framework for enforcing arbitrary nonlinear equality constraints during
 1062 generative sampling.

1063 G Pre-trained Models with Functional Flow Matching

1064 G.1 Flow Matching

1065 Flow Matching (FM) [4, 35, 38] formulates generative modeling as learning a time-dependent velocity
 1066 field $v_\theta(u, \tau)$ that transports a prior sample $u_0 \sim \pi_0$ to a target sample $u_1 \sim \pi_1$. The induced flow
 1067 ϕ_τ satisfies the ODE

$$\frac{d}{d\tau}\phi_\tau(u) = v_\theta(\phi_\tau(u), \tau), \quad \phi_0(u_0) = u_0,$$

1068 and defines a trajectory $u(\tau) = \phi_\tau(u_0)$ connecting π_0 to π_1 . The learning objective minimizes the
 1069 squared deviation between the model field and a known target field \hat{v} , evaluated along interpolants
 1070 $u_\tau = (1 - \tau)u_0 + \tau u_1$. This yields the standard Flow Matching loss:

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{\tau \sim \mathcal{U}[0,1], u_0 \sim \pi_0, u_1 \sim \pi_1} [\|v_\theta(u_\tau, \tau) - \hat{v}(u_\tau, \tau)\|^2], \quad (27)$$

1071 where $\hat{v}(u_\tau, \tau)$ is the true vector field, which is typically unknown.

1072 **Conditional Flow Matching (CFM).** When the conditional velocity field $u_t(u_1) := \partial_\tau \psi_\tau(u_1)$ is
 1073 known in closed form, as under optimal transport, one can minimize the conditional flow matching
 1074 loss:

$$\mathcal{L}_{\text{CFM}}(\theta) = \mathbb{E}_{\tau, u_0, u_1} [\|v_\theta(u_\tau, \tau) - (u_1 - u_0)\|^2], \quad (28)$$

1075 where $u_\tau = (1 - \tau)u_0 + \tau u_1$ is a linear interpolant. This CFM objective forms the basis for most
 1076 recent flow-based generative models due to its tractability and effectiveness.

1077 G.2 Functional Flow Matching

1078 To handle infinite-dimensional generative tasks such as PDE solutions, Functional Flow Matching
 1079 (FFM) [10] extends CFM to Hilbert spaces \mathcal{U} of functions $u : \mathcal{X} \rightarrow \mathbb{R}$. The FFM flow ψ_τ satisfies

$$\frac{d}{d\tau}\psi_\tau(u) = v_\theta(\psi_\tau(u), \tau), \quad u_0 \sim \mu_0,$$

1080 where μ_0 is a base measure over function space. Under regularity assumptions, FFM minimizes an
 1081 analogous loss over the path of measures:

$$\mathcal{L}_{\text{FFM}}(\theta) = \mathbb{E}_{\tau, u_0, u_1} \left[\|v_\theta(u_\tau, \tau) - (u_1 - u_0)\|_{L^2(\mathcal{X})}^2 \right], \quad (29)$$

1082 where $u_\tau = (1 - \tau)u_0 + \tau u_1$ and the L^2 -norm is used to evaluate function-valued velocities over
 1083 the domain \mathcal{X} . This loss generalizes CFM to the functional setting, and serves as the pretraining
 1084 objective for PCFM in this work.

1085 H Sampling Setups for PCFM and Other Baselines

1086 For all comparisons, we adopt the explicit Euler integration scheme unless otherwise specified.
 1087 We use 100 Euler update (flow matching) steps for the simpler heat problem and 200 steps for
 1088 Navier-Stokes, Reaction-Diffusion, and Burgers (IC or BC).

1089 **Vanilla FFM.** We perform unconstrained generation by integrating the learned flow vector field
 1090 using explicit Euler steps, following the standard flow matching procedure [4, 10]. At each step, the
 1091 model applies the forward update $u_{t+\Delta t} = u_t + \Delta t v_\theta(t, u_t)$ without any constraint enforcement.

1092 **PCFM (Ours).** PCFM performs explicit Euler integration combined with constraint correction at
 1093 every step. We apply 1 Newton update to project intermediate states onto the constraint manifold.
 1094 Additionally, we optionally apply guided interpolation with $\lambda = 1.0$, step size 0.01, and 20 refinement
 1095 steps to further refine interpolated states, although we use $\lambda = 0$ by default (no interpolation guidance)
 1096 for computational efficiency (see Appendix L.2 for further ablations on λ). Interpolation guidance is
 1097 only applied for the heat equation case to obtain results in Table 3. For Navier-Stokes, we follow the
 1098 stochastic interpolant idea and randomize noise over batches [36] (i.e., different noise samples over
 1099 batches). Finally, after all flow matching steps, we apply a final projection on the solution to enforce
 1100 the constraints.

1101 **ECI.** We follow the ECI sampling procedure introduced by Cheng et al. [11] where it performs
 1102 iterative extrapolation-correction-interpolation steps as well as several mixing and noise resampling
 1103 steps throughout the trajectory. For the simpler heat equation case, we did not do mixing and noise
 1104 resampling. For more challenging PDEs (Navier-Stokes, Reaction-Diffusion, and Burgers), we apply
 1105 $n_{\text{mix}} = 5$ mixing updates per Euler step, and resample the Gaussian prior every 5 steps. We adopt
 1106 their value enforcements in IC and Dirichlet BC cases, and also constant mass integral enforcement
 1107 in the heat and Navier-Stokes problems, where the periodic boundary conditions lead to trivial linear
 1108 constraints.

1109 **D-Flow.** Following Ben-Hamu et al. [34], D-Flow differentiates through an unrolled Euler integra-
 1110 tion process. Following their paper and the set-up in Cheng et al. [11], we optimize the initial noise
 1111 via LBFGS with 20 iterations and a learning rate of 1.0. The optimization minimizes the constraint
 1112 violation at the final state, requiring gradient computation through the full unrolled trajectory. We
 1113 adopt the adjoint method shipped from torchdiffeq [37] to differentiate through the ODE solver.
 1114 Depending on the problem, we adopt an IC or BC loss as well as a PINN-loss based on the differential
 1115 form of the PDE to form the constraint loss function to be optimized. We use 10 iterations and a
 1116 learning rate of 10^{-2} for the Navier-Stokes dataset to avoid NaNs in the optimization loop.

1117 **DiffusionPDE.** We adopt a gradient-guided sampling method proposed by Huang et al. [9] on the
 1118 same pretrained FFM model. The generation process is augmented with explicit constraint correction
 1119 where a composite loss of IC/BC and PINN [21] (differential form of the PDE), like in the D-Flow
 1120 set-up, is used to guide the vector field at each flow step. We apply a correction coefficient $\eta = 1.0$
 1121 and set the PINN loss weight to 10^{-2} for stable generation. We find that a higher PINN weight or
 1122 higher learning rate lead to unstable generation.

1123 I Batched Differentiable Solver for Nonlinear Constraints

1124 We describe our solver for projecting a predicted sample $u_1 \in \mathbb{R}^n$ onto the nonlinear constraint
 1125 manifold $\mathcal{M} := \{u \in \mathbb{R}^n : h(u) = 0\}$, where $h : \mathbb{R}^n \rightarrow \mathbb{R}^m$. The projection is formulated as the
 1126 constrained optimization problem:

$$\min_{u \in \mathbb{R}^n} \frac{1}{2} \|u - u_1\|^2 \quad \text{subject to} \quad h(u) = 0. \quad (30)$$

1127 The corresponding Lagrangian is:

$$\mathcal{L}(u, \lambda) = \frac{1}{2} \|u - u_1\|^2 + \lambda^\top h(u),$$

1128 with first-order optimality conditions:

$$\nabla_u \mathcal{L}(u, \lambda) = u - u_1 + J(u)^\top \lambda = 0, \quad h(u) = 0,$$

1129 where $J(u) := \nabla h(u) \in \mathbb{R}^{m \times n}$ is the constraint Jacobian.

1130 These yield the full nonlinear KKT system:

$$\begin{cases} u - u_1 + J(u)^\top \lambda = 0, \\ h(u) = 0. \end{cases} \quad (31)$$

1131 **Newton-Based Update.** At iteration k , we linearize the KKT system around $u^{(k)}, \lambda^{(k)}$ and solve
 1132 for updates $\delta u, \delta \lambda$. The full Newton step solves:

$$\begin{bmatrix} I + \sum_{i=1}^m \lambda_i^{(k)} \nabla^2 h_i(u^{(k)}) & J(u^{(k)})^\top \\ J(u^{(k)}) & 0 \end{bmatrix} \begin{bmatrix} \delta u \\ \delta \lambda \end{bmatrix} = - \begin{bmatrix} u^{(k)} - u_1 + J(u^{(k)})^\top \lambda^{(k)} \\ h(u^{(k)}) \end{bmatrix}. \quad (32)$$

1133 Here, the upper-left block contains the Hessian of the Lagrangian:

$$\nabla_{uu}^2 \mathcal{L}(u^{(k)}, \lambda^{(k)}) = I + \sum_{i=1}^m \lambda_i^{(k)} \nabla^2 h_i(u^{(k)}).$$

1134 After solving, we update the primal and dual variables:

$$u^{(k+1)} = u^{(k)} + \delta u, \quad \lambda^{(k+1)} = \lambda^{(k)} + \delta \lambda.$$

1135 This Newton system converges quadratically under standard regularity assumptions (e.g., full-rank
 1136 Jacobian and Lipschitz-continuous second derivatives). In practice, we often omit the second-order
 1137 terms to yield a Gauss–Newton approximation that is more stable and efficient in high-dimensional
 1138 settings.

1139 **Approximate KKT Solve via Schur Complement.** For inference-time projection, we adopt a
 1140 simplified and batched update using the Schur complement [65]. At each iteration, we set $\lambda = 0$, and
 1141 solve only for the primal update. Eliminating δu from Equation (32), we obtain:

$$(JJ^\top) \lambda = h(u), \quad \delta u = -J^\top \lambda. \quad (33)$$

1142 This gives the Gauss–Newton-style update:

$$u \leftarrow u - J^\top (JJ^\top)^{-1} h(u). \quad (34)$$

1143 We iterate this procedure until convergence or until the constraint residual $\|h(u)\|$ falls below a set
 1144 tolerance. The matrix $JJ^\top \in \mathbb{R}^{m \times m}$ is small and typically well-conditioned for local or sparse
 1145 constraints, enabling efficient solves.

1146 Restarting the dual variable with $\lambda = 0$ at each iteration avoids stale gradient accumulation and
 1147 improves numerical stability. This leads to a robust and memory-efficient projection routine that
 1148 supports batched execution and reverse-mode differentiation.

1149 **Batched and Differentiable Implementation.** We implement the solver in a batched and differentiable fashion to support inference across samples. For a batch of inputs $\{u_1^i\}_{i=1}^B$, we evaluate
 1150 Jacobians J^i , residuals $h(u^i)$, and solve the corresponding Schur systems in parallel using vector-
 1151 ized operations and autodiff-compatible backends (e.g., PyTorch with batched Cholesky or linear
 1152 solvers) [67].
 1153

1154 **Computational Complexity.** The per-sample cost includes:

- 1155 • $\mathcal{O}(m^2n)$ for computing J and J^\top ,
- 1156 • $\mathcal{O}(m^3)$ for solving the Schur complement system,
- 1157 • $\mathcal{O}(n)$ for applying the update.

1158 Since typically $m \ll n$, the overall cost scales as $\mathcal{O}(n)$ per sample.

1159 J Evaluation Metrics

1160 We evaluate each method using 512 generated samples for all 1D problems and 100 samples for the
 1161 2D Navier-Stokes problem. For each setting, we use an equivalent number of ground truth PDE
 1162 solutions with a fixed IC or BC configuration used during sampling for direct comparison.

1163 **MMSE and SMSE.** Following Kerrigan et al. [10], Cheng et al. [11], we evaluate generation
 1164 fidelity with mean of the MSE and the standard deviation of the MSE as:

$$\text{MMSE} = \|\mu_{\text{gen}} - \mu_{\text{gt}}\|_2^2, \quad \text{SMSE} = \|\sigma_{\text{gen}} - \sigma_{\text{gt}}\|_2^2,$$

1165 where $\mu_{\text{gen}}, \mu_{\text{gt}}$ denote the mean across generated and true PDE solutions, and $\sigma_{\text{gen}}, \sigma_{\text{gt}}$ the standard
 1166 deviations.

1167 **Constraint Error.** To evaluate physical consistency, we compute the ℓ_2 norm of residuals from
 1168 constraint functions \mathcal{R}_* applied to each sample $\hat{u}^{(n)}$, then average across all N samples:

$$\text{CE}(\ast) = \frac{1}{N} \sum_{n=1}^N \left\| \mathcal{R}_\ast \left(\hat{u}^{(n)} \right) \right\|_2, \quad \ast \in \{\text{IC}, \text{BC}, \text{CL}\}.$$

1169 where the residuals are computed following Appendix D to measure the constraint or conservation
 1170 violation.

1171 **Fréchet Poseidon Distance (FPD).** To assess distributional similarity beyond mean and variance,
 1172 we adopt the Fréchet Poseidon Distance (FPD) introduced in Cheng et al. [11], where we measure
 1173 the Fréchet distance between the hidden state distributions extracted from a pretrained foundation
 1174 model (Poseidon [63]) applied to both generated and true solutions. We pass both generated and true
 1175 solutions through the Poseidon base model (157M parameters) and extract the last hidden activations
 1176 of the encoder to eventually obtain a 784-dimension feature vector for FPD calculation:

$$\text{FPD}^2 = \|\mu_1 - \mu_2\|^2 + \text{Tr} \left(\Sigma_1 + \Sigma_2 - 2(\Sigma_1 \Sigma_2)^{1/2} \right), \quad (35)$$

1177 where μ_1, Σ_1 and μ_2, Σ_2 are the empirical mean and covariance of the Poseidon embeddings from
 1178 the generated and true solutions' distributions, respectively. FPD is computed either per frame $u(x, y)$
 1179 for 2D PDEs and averaged over all frames (across t) or over the full spatiotemporal solution $u(x, t)$
 1180 for 1D PDEs.

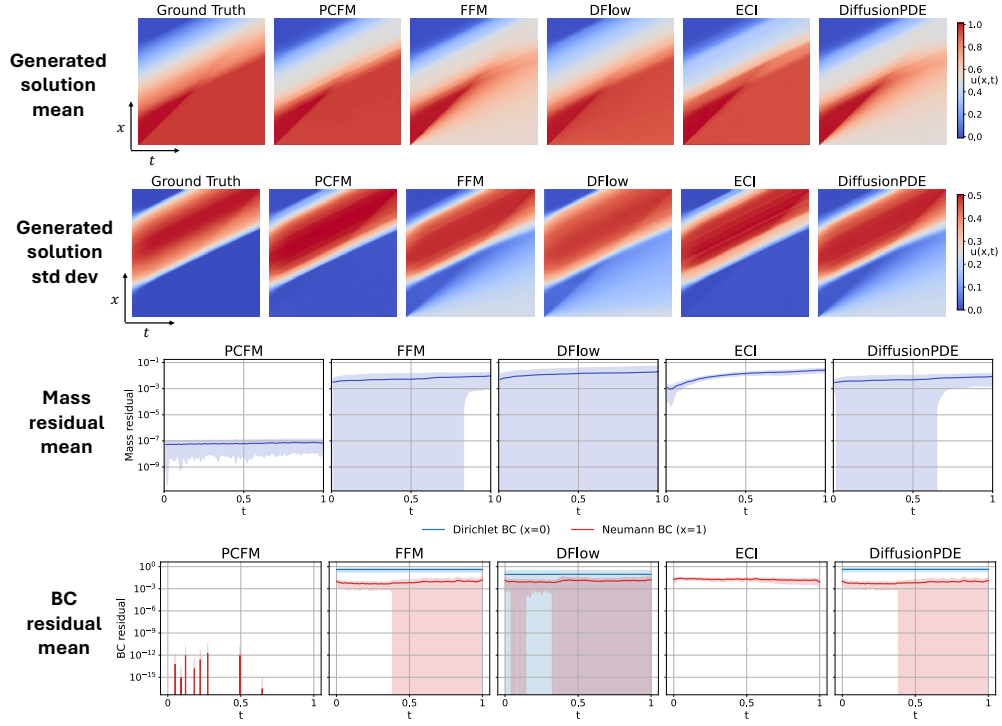


Figure 5: Solution profiles for the Inviscid Burgers equation with fixed BC. We plot the various constraint guidance methods and compare the mean solution profile and standard deviation. While PCFM yields slightly worse MMSE and SMSE and better FPD, it ensures global mass conservation and maintains low constraint errors for both Dirichlet and Neumann BCs over time.

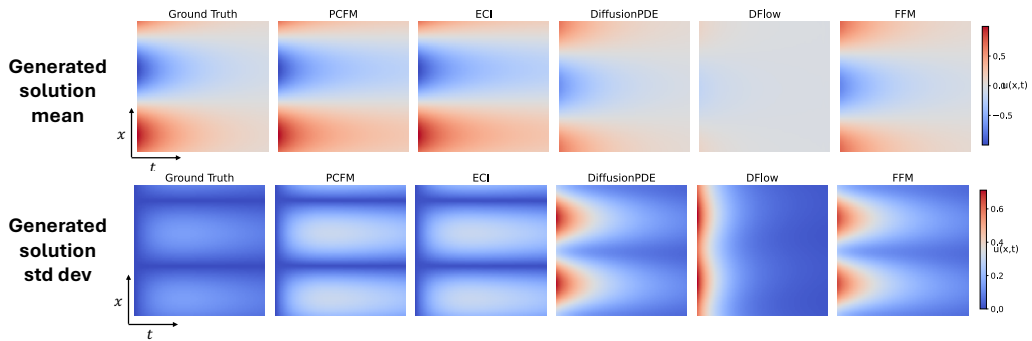


Figure 6: Solution profiles for the Heat equation with fixed IC. We plot the various constraint guidance methods and compare the mean solution profile and standard deviation. PCFM outperforms all other methods by visually being the most similar to the ground truth.

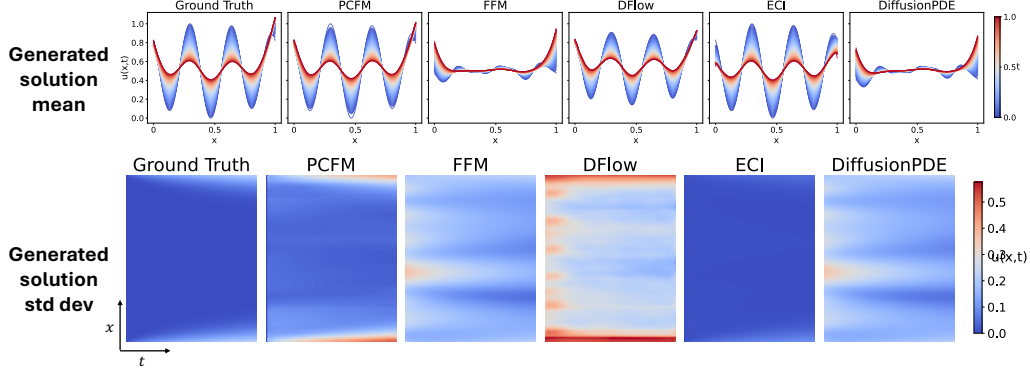


Figure 7: Alternative view of the Reaction-Diffusion equation with fixed IC. We plot the pointwise mean and standard deviation across generated samples for each method. PCFM outperforms all other methods by visually being the most similar to the ground truth.

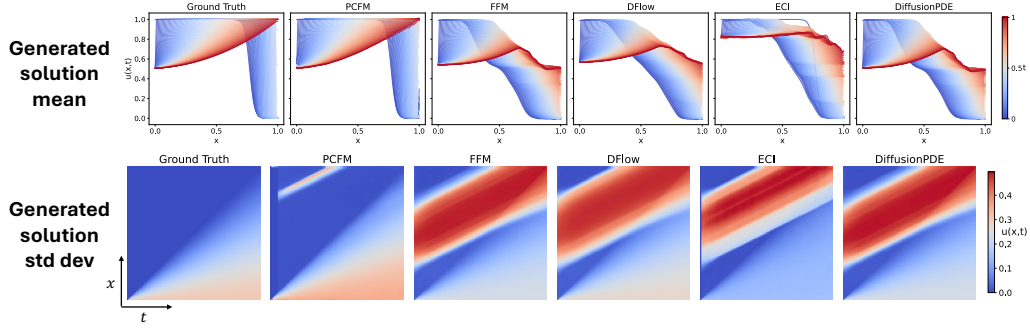


Figure 8: Alternative view of the Inviscid Burgers equation with fixed IC. We plot the pointwise mean and standard deviation across generated samples for each method. PCFM outperforms all other methods by visually being the most similar to the ground truth.

L Further Ablations

L.1 Total Variation (TV) Constraints on the Heat Equation

Total Variation Diminishing (TVD) constraints encode the principle that diffusive systems, such as those governed by the heat equation, smooth spatial fluctuations over time. Mathematically, this is expressed by the fact that the total variation of the solution should not increase in time:

$$\text{TV}(u(t)) := \int \left| \frac{\partial u}{\partial x} \right| dx \quad \text{is non-increasing in } t.$$

To enforce this property in a data-driven or generative setting, we adopt a hard constraint that relates the total variation at final time T to the total variation at the initial time $t = 0$. Specifically, we impose the condition:

$$\text{TV}(u_T) = \gamma \cdot \text{TV}(u_0),$$

where $\gamma \in (0, 1)$ is a decay factor that can either be fixed (e.g., $\gamma = 0.5$) or estimated from unprojected samples. This constraint can be implemented using discrete spatial differences as:

$$\text{TV}(u_j) \approx \sum_{i=0}^{n_x-2} |u_{i+1,j} - u_{i,j}|,$$

applied at selected time slices (typically $t = 0$ and $t = T$).

Table 4: Test metrics on the Heat Equation dataset with TVD constraints. Lower is better for all metrics.

Metric	PCFM	ECI	DiffusionPDE	D-Flow	FFM
MMSE / 10^{-2}	0.684	0.697	4.49	1.97	4.56
SMSE / 10^{-2}	0.962	0.973	3.93	1.14	3.51
CE (IC) / 10^{-2}	0	0	599	102	579
CE (CL) / 10^{-2}	0	0	2.06	64.8	2.11
FPD	1.28	1.34	1.70	2.70	1.77
Power MSE	488.42	500.12	1420.92	3996.11	932.20

We encode the constraint as a differentiable function:

$$h_{\text{TV}}(u) := \text{TV}(u_T) - \gamma \cdot \text{TV}(u_0) = 0.$$

This constraint is compatible with mass conservation and initial condition enforcement, and can be combined with other physically informed conditions such as energy decay or curvature regularity. In practice, we observe that enforcing TVD constraints reduces spurious oscillations in the generative output and improves structural fidelity without significantly altering statistical accuracy.

Power MSE. To further quantify the effect of this constraint, we introduce the *Power MSE* metric, which measures the deviation in spatial frequency content between the generated and ground truth solutions. Formally, for a model’s mean output $\bar{u}(x, t)$, we compute the spatial power spectrum $|\hat{u}(k, t)|^2$ and average over time to obtain a per-frequency energy profile. The Power MSE is then defined as:

$$\text{Power MSE} := \frac{1}{n_x} \sum_k \left(|\overline{\hat{u}_{\text{gen}}(k)}|^2 - |\overline{\hat{u}_{\text{ref}}(k)}|^2 \right)^2.$$

This metric captures discrepancies in frequency modes and is especially sensitive to unphysical sharpness or over-smoothing. Variants of power-spectrum-based error metrics have been used in turbulence modeling [68], climate downscaling [69], and Fourier neural operators training [22]. In our setting, we find that Power MSE effectively complements pointwise metrics (e.g., MMSE, CE), as it evaluates structural fidelity in the spectral domain. As shown in Table 4, this metric reveals how enforcing a TVD constraint not only improves calibration but also leads to closer alignment in frequency space.

Effect of TVD Constraint. To isolate the benefit of total variation control, we mirror the **ECI** setting by setting the constraint penalty weight to zero and enforcing only initial condition and mass conservation constraints. When we additionally include the TVD constraint ($\gamma = 0.3$) in our **PCFM** model (with no penalty-based regularization), we observe a consistent improvement in structural fidelity. This is most evident in the Power MSE metric, where PCFM achieves the lowest error among all baselines (488.42 vs. 500.12 for ECI), indicating better alignment with the true spatial energy distribution. We also see gains in MMSE and SMSE, without introducing constraint violations (e.g., CE remains zero). This confirms that the TVD constraint acts as a meaningful structural prior even in the absence of learned penalties, improving generative realism without sacrificing calibration or efficiency.

L.2 Effect of Relaxed Constraint Correction with Flow Matching Steps

Figure 9 shows the effect of the relaxed constraint penalty weight λ on the generation error, measured via MMSE and SMSE, for the Reaction-Diffusion dataset. We evaluate on a 128×100 spatial-temporal grid, as described in Appendix B.3, and solve the relaxed optimization objective in Equation (6) using the Adam optimizer.

With only 10 flow steps (left), increasing λ significantly improves performance, as the relaxed constraint correction effectively compensates for the coarser integration of the flow. In contrast, with 100 flow steps (right), the flow is more precise, and additional penalty yields only marginal benefit. This illustrates that relaxed correction is particularly valuable when inference is constrained to a small

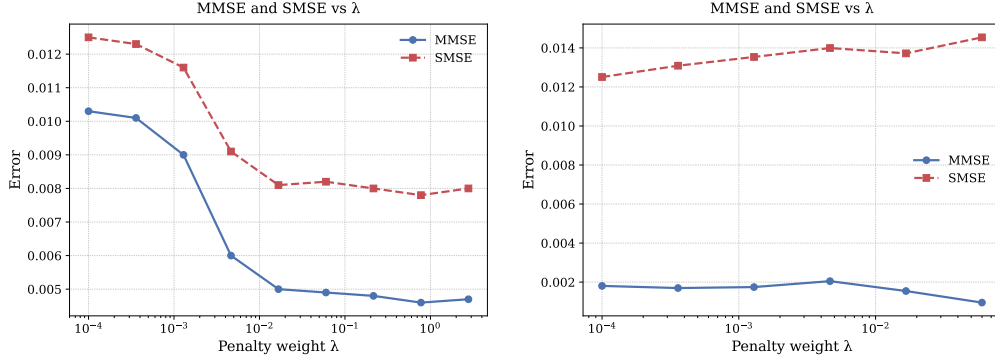


Figure 9: Effect of penalty weight λ on MMSE and SMSE for the Reaction-Diffusion dataset. Left: 10 flow matching steps. Right: 100 flow matching steps.

number of steps—for instance, in scenarios where evaluating the vector field v_θ is computationally expensive.

However, overly large values of λ can harm performance by distorting the reverse update and breaking alignment with the OT interpolant (see Proposition 3.1), as discussed in Equation 5. Hence, choosing λ requires balancing constraint enforcement with consistency along the learned generative path.