

Supplementary Material

S1 Software and Computational Resources

For all baseline SBI methods, we use the sbi toolbox [29], for the Simformer baseline we use the publicly available code from Gloeckler et al. [30]. We use an optimized solver to solve the Darcy Flow PDE [38].

We use various compute resources for the different experiments. For each experiment, we run the training and evaluation for each method, for each simulation budget, and for each of the three random seeds separately. We provide information on the maximum wall-clock time of these runs for each experiment.

For the Linear Gaussian and SIRD experiments, we perform our experiments on Nvidia RTX 2080ti GPU nodes. Both simulators have negligible wall-clock costs on these GPU nodes. All training runs finished within 2 hours of wall-clock time for all methods and all simulation budgets.

The Darcy flow experiment required GPUs with higher VRAM to accommodate the large ($\approx 16k$ dimensional) parameters and observations. We performed these experiments on Nvidia A100 GPUs for both training and evaluation. Each training run for each method at all budgets was complete within 3 hours of wall-clock time.

For the mass balance rates experiment, we perform training and evaluation on CPU, namely Intel Xeon Gold 16 cores, 2.9GHz. We perform this experiment on CPU as the main computation cost is in running the simulations for the predictive MSE check, and the implementation of the model is not accelerated by use of GPUs. The exact simulation costs are described in Redacted [39]. We perform 1000 simulations to calculate the mean predictive MSE in each training run. Each training and evaluation run for all method was completed within 24 hours of wall-clock time.

S2 Evaluation details

We here describe more details about our evaluation procedures. We evaluate on a heldout test set $\{(\theta_j^o, l_j^\theta, \eta_j^o, x_j^o, l_j^x)\}_{j=1}^{N_{\text{test}}}$, where N_{test} is the number of test simulations. Given an approximate posterior distribution $q_{l^\theta}^\phi(\theta, \eta \mid x, l^x)$ and a test observation (x_j^o, l_j^x) , we draw N_{post} posterior samples $(\theta_{ij}, \eta_{ij}) \sim q_{l^\theta}^\phi(\theta, \eta \mid x_j^o, l_j^x)$. In the case where no vector-valued parameters η are present, they can be omitted. Similarly, for methods which do not explicitly use the positions l^θ, l^x (e.g. FNOPE (fix)), the positions can be omitted, as we do not apply these methods to tasks where we consider arbitrary discretizations.

We report the average and standard error over all N_{test} test simulations.

S2.1 Sliced Wasserstein Distance

We calculate the sliced Wasserstein(-2) distance (SWD) [27], with 50 random projections.

S2.2 Simulation-based Calibration Error of Diagonal

Simulation-based calibration (SBC) [28] is a standard measure of the calibration of approximate posterior distributions (in terms of over- or underconfidence). We obtain ranks r_{ij} for each sample (θ_j^o, x_j^o) in the test set using SBC with the 1-dimensional marginal distributions used as the reducing functions. That is, for each of the dimensions i of θ , the rank r_{ij} is an integer in $(1, N_{\text{post}} + 1)$. This results in $N_{\text{test}} \times d_\theta$ ranks. The cumulative distribution function of ranks is therefore

$$\text{CDF}(\alpha) = \frac{1}{N_{\text{test}} \cdot d_\theta} \sum_j \mathbb{I}[r_j / N_{\text{post}} < \alpha].$$

The SBC Error of Diagonal (SBC EoD) is then the mean absolute distance between this cumulative distribution and the cumulative distribution function of a uniform distribution,

$$\text{SBC EoD} = \int_0^1 |\text{CDF}(\alpha) - \alpha| d\alpha.$$

878 In contrast to the SBC area under the curve (SBC AUC), the EoD will detect poor calibrations
 879 for posteriors that are overconfident at low confidence levels α , and underconfident at high α (or
 880 vice-versa).

881 S2.3 Predictive MSE

882 To calculate the MSE for posterior predictive samples, we run for each posterior sample (θ_{ij}, η_{ij}) ,
 883 and each true observation x_j^o , the simulator $x_{ij} \sim p_{l_j^x}(x \mid \theta_{ij}, l^{\theta}, \eta_{ij})$. We then compute the average
 884 mean square error of the simulation x_{ij} to the corresponding observations x_j^o ,

$$\text{MSE} = \frac{1}{N_{\text{test}} N_{\text{post}}} \sum_{i=1, j=1}^{N_{\text{test}}, N_{\text{post}}} \frac{1}{|l_j^x|} \|x_{ij} - x_j^o\|_{L^2}^2,$$

885 where $|l_j^x|$ is the number of points in the discretization l_j^x and therefore the dimensionality of x_j^o .
 886 We use this metric since the simulators considered in this work correspond to (unknown) unimodal
 887 likelihood functions—a correctly estimated posterior will produce simulations clustered around the
 888 true observation. We opt for this metric to quantify predictive performance due its clear interpretability.
 889 However, for multimodal likelihood functions, this metric can be replaced with a scoring rule.

890 S2.4 Posterior Log Probability

891 For the Darcy Flow task, we additionally report the posterior log-probability of the true parameters
 892 [\[31\]](#), normalized by the number of pixels:

$$\text{log-probability per pixel} = \frac{1}{|l_j^{\theta}|} \log q_{l_j^{\theta}}^{\phi}(\theta_j^o \mid x_j^o, l_j^x).$$

893 For the spectral methods NPE/FMPE(spectral), we cannot directly compute the posterior-log-
 894 probabilities, as we can only compute the posterior-log-probabilities of the first M modes of the
 895 spectral decomposition of the ground truth parameters θ_j^o . However, by discarding the information
 896 of the higher modes, we remove the information which these baseline methods cannot capture, thus
 897 biasing the resulting log-probabilities in favor of these baselines.

898 S3 FNOPE details

899 S3.1 Kernel lengthscale heuristic

900 The spectral density of samples from a Gaussian Process with a square exponential kernel of
 901 lengthscale l is stated in Sec. [3.1](#) as

$$P(f) = (2\pi l^2)^{d_{\theta}/2} \exp(-2\pi^2 l^2 f^2).$$

902 This is also a Gaussian density, and trivially we see that the full spectral power is

$$\bar{P} = \int_{\mathbb{R}^{d_{\theta}}} P(f) df = 1$$

903 We consider discretizations normalized to $[0, 1]$ in each dimension, and so the power contained in the
 904 first M spectral modes, \bar{P}_M , corresponds to the above integral within the domain $\|f\|_{\infty} \leq M/2$, i.e.
 905 where all the components of f are within $[-M/2, M/2]$. Therefore \bar{P}_M simplifies to the product of
 906 the Gaussian integrals

$$\begin{aligned} \bar{P}_M &= \left(\int_{-M/2}^{M/2} (2\pi l^2)^{1/2} \exp(-2\pi^2 l^2 f_i) df_i \right)^{d_{\theta}} \\ &= \left(\text{erf} \left[\frac{\pi l M}{\sqrt{2}} \right] \right)^{d_{\theta}} \\ &= \left(\text{erf} \left[\frac{M \sqrt{2}}{M/2 + 1} \right] \right)^{d_{\theta}}, \end{aligned}$$

where erf is the Gauss error function, and in the last line we substituted our heuristic $l = \frac{2}{\pi(M/2+1)}$. This value saturates the error function and produces values very close to 1. For example, for the Darcy Flow example ($d_\theta = 2$), we set $M = 32$. The resulting spectral power in the first 32 modes is $\bar{P}_{32} \approx 0.9997$. While individual samples from the Gaussian process can result in discrete Fourier transforms where this spectral property is not fulfilled, it is clear that the majority of the spectral power will be contained in the first M modes for all samples.

913 S3.2 Flexible discretization

We provide further detail of the data augmentation scheme introduced in Sec. 3.2. First, we describe why this is necessary despite the use of the non-uniform fast fourier transform (NUDFT). The NUDFT is applied as a matrix multiplication, $\Theta = V(l^\theta)\theta$, where Θ is a vector containing the first M spectral components of θ , and V is the discretization-dependent transformation matrix. The inverse NUDFT is similarly implemented as a matrix multiplication $\theta = \bar{V}^\top(l^\theta)\Theta$, where \bar{V}^\top is the conjugate transpose of V . This approach enables the computational efficiency of the NUDFT, as the exact inverse matrix does not need to be computed at runtime. However, \bar{V}^\top is only an approximate inverse of V , with the approximation error increasing for increasing non-uniformity of the discretization.

Consider the common case where the simulation dataset S (Sec. 3) provides parameters θ_i and observations x_i always discretized on the same, uniform simulation domain. Without data augmentation, we always apply the NUDFT and its inverse without approximation error. However, if we wish to condition a posterior on x^o measured at some non-uniform discretization l^x , then the NUDFT and its inverse will produce some error, which was unseen during training. This could lead to unpredictable, out of distribution errors at evaluation time. By explicitly passing the positions l^θ and l^x to the network, as well as augmenting them to ensure the network is not always applied on uniform discretizations during training, we give FNOPE capacity to learn to counteract these approximation errors.

Masking We define a uniform distribution over the binary mask vectors with a fixed number of nonzero entries, N_{ds} . Suppose we are given a simulation $(\theta, l^\theta, x, l^x)$, where θ, l^θ consist of N_θ points, and x, l^x consist of N_x points. We construct two random binary masks, $\mathbf{M}^\theta \in \{0, 1\}^{N_\theta}$, $\mathbf{M}^x \in \{0, 1\}^{N_x}$, each with exactly N_{ds} nonzero entries. We then remove the corresponding elements of θ, l^θ where \mathbf{M}_θ is zero, and similarly remove the corresponding elements of x, l^x where \mathbf{M}_x is zero. For a minibatch, of simulations, we independently sample the masks $\mathbf{M}_i^\theta, \mathbf{M}_i^x$ for each simulation. If $N_{\text{ds}} > N_\theta$ or $N_{\text{ds}} > N_x$, we leave the corresponding value and position vector unchanged. The value of N_{ds} used in our work is reported for each experiment in Appendix S6.

Positional noise We additionally add small, independent gaussian noise $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$ to each point in l^θ and l^x in the unmasked positions. This reduces generalization error for simulation datasets where the discretization of θ and x is fixed. The value of σ^2 can be set according to the spacing of the discretization. In our experiments, we always normalize the simulation domain to $[0, 1]$ in all dimensions, and set $\sigma = 10^{-3}$.

945 S3.3 Fixed discretization

For applications with uniform grids, we provide FNOPE (fix). Here, we use the FFT instead of the NUDFT in the FNO blocks to transform the data from physical to spectral space and back. In addition, we do not mask any parameters during training and do not add any positional noise. We expect this method to have improved performance on uniform grids as we do not introduce additional noise through the data augmentation process described above.

Another potential advantage of FNOPE (fix) is its computational efficiency. Consider the case of a parameter discretized uniformly in D dimensions, with L points per dimension, leading to a total of L^D points. The computational cost of computing the spectral decomposition of the parameters using the FFT is $O(L^D \log L^D) = O(DL^D \log L)$. Assuming the maximum number of modes in each dimension modeled by the FNO is M , this leads to M^D total modes to compute. Therefore, the computational cost of the frequency-limited NUDFT is $O(M^D L^D)$. For one-dimensional domains, the NUDFT may well be faster to compute than the FFT. However, for higher dimensions, the NUDFT

scales exponentially with both M and L . This discrepancy increases with both the dimensionality of the domain, and the number of modes M modeled by the FNO blocks.

S3.4 Additional Parameters

The naïve extension of the FMPE objective as stated in Sec. 3.3 is to minimize the L^2 loss $\|v^\phi - u_t\|^2$, where $v^\phi = [v_{l^\theta}^\phi, v_\eta^\phi]$ and $u_t = [u_t(\xi_t|\theta), u_t(z_t|\eta)]$. However, the scale of the loss for the continuous parameters, $\|v_{l^\theta}^\phi - u_t(\xi_t|\theta)\|^2$ varies with the number of points in the discretization l^θ , which we denote N_θ . To ensure that the loss is balanced for the function- and vector-valued parameters, we in practice add their L^2 losses, normalized by their respective vector dimensionalities. That is, we minimize

$$\mathbb{E} \frac{1}{N_\theta} \|v_{l^\theta}^\phi - u_t(\xi_t|\theta)\|^2 + \frac{1}{N_\eta} \|v_\eta^\phi - u_t(z_t|\eta)\|^2,$$

where N_η is the dimensionality of η . The expectation is over the same random variables as for the statement of the loss \mathcal{L}_2 in Sec. 3.3

S4 Baseline Methods

S4.1 Spectral preprocessing

For spectral NPE/FMPE we first apply a Fourier transformation to the parameters, take the first M Fourier modes and expand these M complex values to a real vector of dimension $2M$ representing the real and imaginary parts (for two dimensional data this results in $2M^2$ parameters). After inferring the posterior of the parameters in Fourier space and sampling from it, we apply the inverse Fourier transform to get samples in the spatial domain.

For the one dimensional problems (Linear Gaussian and Ice Shelf) we first pad the data by replicating the first/last value and perform a real FFT with `torch.fft.rfft`. We then use the first M fourier components and expand these complex numbers to a real tensor of dimension $2M$, which we use as input to NPE/FMPE. For the Linear Gaussian and Ice Shelf we use $M = 50, 10$, respectively. We then revert this process for samples from the posterior with `torch.fft.irfft` with the corresponding settings.

For the two dimensional Darcy flow, we use the two dimensional FFT implemented in *pytorch* on the padded data (in mode ‘replicate’). We then center the frequencies before cropping to the first M Fourier components in both dimensions, and expanding it to a real tensor of dimension $2M^2$. For posterior samples we again revert this process with the corresponding settings.

S4.2 NPE (spectral)

For NPE (spectral) we infer the posterior over the coefficients of the first M Fourier modes following the spectral preprocessing described above. We use NPE [3] with normalizing flows. We do not apply spectral preprocessing to the observations but pass raw observations through an embedding net as it is common practice in NPE.

S4.3 NPE (raw)

For the mass balance experiment (Sec. 4.5), we also compare to the approach of Redacted [39], which we refer to as NPE (raw). This approach infers the mass balance parameters on a fixed discretization of 50 gridpoints. The authors use a Neural Spline Flow with 5 transformations, two residual blocks of 50 hidden units each, ReLU nonlinearities and 10 bins. The embedding net used to embed the 441-dimensional observation is a CNN with two convolutional layers of kernel size 5, with ReLU activations and max pooling of kernel size 2. The convolutional layers are followed by two linear layers with 50 hidden units and output dimension 50. The same settings are used in the 500-dimensional experiment (Appendix S7). Training is performed with a batch size of 200 and an Adam optimizer with learning rate of 0.0005.

1001 S4.4 FMPE (spectral)

1002 As with NPE (spectral), in this approach we apply spectral preprocessing to the parameters, and
1003 infer the coefficients of the top M Fourier modes, but process the observations directly using an
1004 embedding net. We use MLPs to estimate the flows, as in Wildberger et al. [9], as implemented in the
1005 sbi toolbox [29]. This implementation also uses the rectified flow [18] objective for FMPE, and we
1006 use independent Gaussian noise as the noise distribution. The flow networks are conditioned on time
1007 by concatenating the time to the inputs.

1008 S4.5 FMPE (raw)

1009 In this approach, we infer the parameters directly on a fixed discretization of the domain. We again use
1010 embedding nets to encode the observations, and MLPs to learn the flows. As with FMPE (spectral),
1011 we use a rectified flow objective, and independent Gaussian noise as the noise distribution, as in the
1012 sbi toolbox. The flow networks are conditioned on time by concatenating the time to the inputs.

1013 S4.6 Simformer

1014 For the SIRD experiment, we apply Simformer with the same settings as in Gloeckler et al. [30]. That
1015 is, we use a transformer model with a token dimension of 50, 8 layers, and 4 heads. The widening
1016 factor is 3, and the training was performed with a batch size of 1000 and an Adam optimizer. We
1017 train Simformer to learn all conditionals, and so uniformly draw between the posterior, joint, and
1018 likelihood masks, as well as two random masks drawn from Bernoulli distributions with $p = 0.3$
1019 and $p = 0.7$ respectively. For both SIRD experiments, where we evaluate on 20 and 40 time points
1020 respective, we use the same Simformer model which is trained using 20 randomly sampled time
1021 points.

1022 S5 Simulators

1023 S5.1 Linear Gaussian model

1024 The Gaussian Simulator is inspired by Lueckmann et al. [31], but instead of a 10 dimensional
1025 Gaussian distribution with independent dimensions we expanded the problem to 1000 dimensions
1026 and use a Gaussian Process prior (see below). Draws from the simulator are still drawn independently
1027 per dimension as $x \sim \mathcal{N}(\theta, \sigma^2 \mathbf{I})$, where $\sigma^2 = 0.1$ (as in [31]).

1028 **Prior** The prior is defined as a Gaussian process \mathcal{GP} on $[0, 1]$ with an equidistant discretization
1029 with 1000 timepoints. A draw from the prior is therefore defined as $\theta \sim \mathcal{GP}(0, k(\cdot, \cdot))$, where k is the
1030 squared exponential kernel, $k(t, t') = \exp(-\frac{(t-t')^2}{2l^2})$. We set the lengthscale $l = 0.05$ and variance
1031 to 1.

1032 **Evaluation parameters** The results for Fig. 3 is based on 100 observations and 1000 posterior
1033 samples for each observation.

1034 S5.2 SIRD model

1035 Similar to Gloeckler et al. [30] we extend the SIRD (Susceptible, Infected, Recovered, Deceased)
1036 model to have a time-dependent contact rate. Compared to the classical SIR framework the model
1037 additionally incorporates a Deceased (D) population. Similar models were explored by Chen et al.
1038 [33], Schmidt et al. [34]. This addition is important for modeling diseases with significant mortality
1039 rates. The SIRD model, including a time-dependent contact rate $\beta(t)$, is defined by the following set

1040 of differential equations:

$$\begin{aligned}\frac{dS}{dt} &= -\beta(t)SI, \\ \frac{dI}{dt} &= \beta(t)SI - \gamma I - \mu I, \\ \frac{dR}{dt} &= \gamma I, \\ \frac{dD}{dt} &= \mu I.\end{aligned}$$

1041 Here, S , I , R and D are the susceptible, infected, recovered, and deceased population, $\beta(t)$ is the
1042 time dependent contact rate, and γ and μ are the recovery and mortality rates among the infected
1043 population. We simulate on a dense uniform grid of 100 time points for parameters and observations.
1044 The simulations are additionally contaminated by an observation noise model, which is described by
1045 a log-normal distribution with mean $S(t)$ and standard deviation $\sigma = 0.05$.

1046 **Prior** We impose the same prior as in Gloeckler et al. [30]: the global variables γ and μ are drawn
1047 from a Uniform distribution, $\gamma, \mu \sim \text{Unif}(0, 0.5)$. For the time-dependent contact rate we define a
1048 Gaussian process prior which is further transformed by a sigmoid function to ensure that $\beta(t) \in [0, 1]$
1049 for all t . For the Gaussian process we use a RBF kernel k defined as $k(t, t') = \exp(-\frac{\|t-t'\|^2}{2\cdot 7^2})$.

1050 **Evaluation parameters** MSE as well as SBC EoD is based on 100 observations with 1000 posterior
1051 samples each (Fig. 4c and Fig. S1c). To calculate the posterior predictive, we sample the initial
1052 condition $I(0)$ from the Simformer prediction for both methods, as opposed to the prior defined
1053 above, to match the setting of Gloeckler et al. [30].

1054 S5.3 Darcy Flow

1055 Details for the Darcy model are already given in the main text. We additionally scale the log-
1056 permeability a by the scale factor of 1000 before taking the exponential - this results in permeabilities
1057 which produce sufficiently variable solutions using the Darcy flow simulator, and the permeabilities
1058 on the same scale as reported in Lim et al. [37]. The simulation output is additionally corrupted by an
1059 independent Gaussian observational noise per pixel $\zeta_i \sim \mathcal{N}(0, \sigma_i^2)$. We set $\sigma_i = \mathbb{E}[u_i^2/30]$, where
1060 the expectation is per simulation batch, resulting in a signal to noise ratio (SNR) of 30.

1061 **Evaluation Parameters** All metrics are calculated over a test set of 10 observations (Fig. 5b-d).
1062 For MSE and SBC EoD, and 100 posterior samples were used for each observation and for each
1063 method. Finally, for SBC EoD, we use a subset of 50 pixels of the full 129×129 as the marginals
1064 used for the reducing functions. The same pixels were used across all methods and all observations.

1065 S5.4 Mass balance rates of Antarctic Ice Shelves

1066 We use the same simulator as described in Redacted [39]. This model takes in spatially varying
1067 surface accumulation rates $\dot{a}(l)$, which are related to the basal melt rates $\dot{b}(l)$ through the total balance
1068 condition $\dot{m}(l) = \dot{a}(l) - \dot{b}(l)$. $\dot{m}(l)$ is known and fixed across all simulations.

1069 The layer prediction model consists of a set of isochronal layer with prescribed thicknesses
1070 $\{h_1(l), h_2(l), \dots, h_K(l)\}$ such that $\sum_{k=1}^K h_k(l) = h(l)$, where $h(l)$ is the known and fixed to-
1071 tal ice shelf thickness. At each time step, the thickness of the layers are simultaneously updated
1072 through an advection equation,

$$\frac{\partial h_k}{\partial t} = -\nabla \cdot (h_k u),$$

1073 where $u(l)$ is the known velocity profile of the ice shelf which is fixed across simulations. Additional
1074 layers are added at the top of the ice shelf and removed from the bottom according to $\dot{a}(l)$, $\dot{b}(l)$
1075 accordingly. The noise model approximates the observation noise of the radar measurement given an
1076 assumed density profile of the ice shelf. At the final timestep T , the layer that mostly closely matches

the ground truth observation x^o according to the L^2 norm is selected as the simulator output. Full details are described in Redacted [39].

Prior The prior is defined over the accumulation rate parameter $\dot{a}(l)$, and is motivated by physical observations at Ekström ice shelf. Prior samples are drawn using

$$\dot{a}(l) = \sigma_{\text{sc}} \dot{\alpha} + \mu_{\text{off}}, \quad (4)$$

where $\mu_{\text{off}} \sim \mathcal{N}(0.5, 0.25^2)$, $\sigma_{\text{sc}} \sim \mathcal{U}([0.1, 0.3])$, and $\dot{\alpha}$ is drawn from a Gaussian Process with a unit-variance, zero-mean Matérn- ν kernel of lengthscale 2500 and $\nu = 2.5$.

Evaluation Parameters For SBC EoD, as well as the predictive MSE on synthetic test simulations, we use 100 test observations and sample 10 posterior samples for each observations, for each method. The real test data consists of one field observation (shown in Fig. 6a-b), and the posterior predictive was estimated using 1000 posterior samples.

S6 Experimental details

S6.1 Linear Gaussian

For all the baseline methods, we train the networks using an Adam optimizer with a learning rate of 0.0001, and a batch size of 200. For NPE/FMPE (spectral), we use 50 modes, leading to 100 parameters to learn, and a pad width of 200 for the spectral preprocessing (Appendix S4.1). For NPE (spectral) the density estimator is a Neural Spline Flow (NSF) with 2 residual blocks with 50 hidden dimensions each, 5 transforms, with RELU activations. For FMPE (spectral), we use an MLP with 5 linear layers with 64 hidden dimensions to estimate the flow, with ELU activations. In both cases, we embed the 1000 dimensional observation into a 40-dimensional vector using an MLP with 2 layers and 50 hidden units and RELU activations. For FMPE (raw), we use an MLP with 5 layers and 64 hidden features, with ELU activations.

For FNOPE and FNOPE (fix) we use 50 Fourier modes for the FNO blocks. We use 5 FNO blocks with 16 channels, while the context is embedded into 8 channels. We train for a maximum of 500 epochs with an early patience of 50. We used a training batch size of 512 and a learning rate of 0.001. For FNOPE, we use 4 channels each for the positional and time embeddings and the target gridsize $N_{\text{ds}} = 256$ (for FNOPE (fix), no positional embedding is included). All nonlinearities are GELUs.

The overall number of trainable parameters for each network in this experiment is summarized in Table S1.

Table S1: **Linear Gaussian Network sizes.** The number of trainable parameters for each of the methods in the Linear Gaussian task.

Method	# Trainable Params
FNOPE	109,797
FNOPE (fix)	108,581
NPE (spectral)	566,590
FMPE (spectral)	34,212
FMPE (raw)	149,032

S6.2 SIRD

For FNOPE we use 32 Fourier modes for the FNO blocks. We use 5 FNO blocks with 16 channels, while the context is embedded into 8 channels. We train for a maximum of 1000 epochs with an early patience of 50. We use a training batch size of 200 and a learning rate of 0.001. The discretization positions and flow times are embedded into 4 channel dimensions each, and the target gridsize $N_{\text{ds}} = 40$. This experiment additionally included vector-valued parameters in \mathbb{R}^2 . These are embedded into a 16-dimensional vector using a 1-layer MLP with a hidden dimension of 64. The flow for the vector-valued parameters is estimated using a 1-layer MLP with a hidden dimension of 64. The spectral decomposition of the output of the FNO blocks, as well as the spectral decomposition

1114 of the observation, are embedded into a 32-dimensional vector and concatenated to the input of the
1115 MLP. All nonlinearities were GELUs.

1116 The training hyperparameters for simformer are described in [S4.6](#).

1117 **S6.3 Darcy Flow**

1118 For all the baseline methods, we train the network using an Adam optimizer with a learning rate of
1119 0.0001, and a batch size of 200. For NPE/FMPE (spectral), we use 16 modes, leading to $2 \times 16^2 = 512$
1120 parameters to learn, and a pad with a width of 20 in each dimension for the spectral preprocessing
1121 (Appendix [S4.1](#)). For NPE (spectral) the density estimator is a NSF with 2 residual blocks with 50
1122 hidden dimensions each, 5 transforms, with RELU activations. For FMPE (spectral), we use an MLP
1123 with 8 layers with 256 hidden dimensions to estimate the flow, with ELU activations. For FMPE
1124 (raw), we use an MLP with 8 layers and 256 hidden features, with ELU activations. All baseline
1125 methods embed the observation with a CNN embedding net into a 100-dimensional vector using
1126 4 convolutional layers with kernel size 5 followed by max pooling of kernel size 2, followed by a
1127 4-layer MLP with 100 hidden units, with RELU nonlinearities throughout.

1128 For FNOPE and FNOPE (fix) we use 32 Fourier modes for the FNO blocks. The network is made
1129 of 5 FNO blocks with 32 channels, while the context is embedded into 32 channels. We train for
1130 a maximum of 300 epochs with an early patience of 50. We use a training batch size of 200 and a
1131 learning rate of 0.0005. For FNOPE, we set the target gridsize $N_{ds} = 2048$. The architecture includes
1132 8 channels for positional embedding, and 8 channels for time embedding. All nonlinearities are
1133 GELUs.

1134 **S6.4 Mass balance rates of Antarctic Ice Shelves**

1135 For all the baseline methods, we train the network using an Adam optimizer with a learning rate
1136 of 0.0001, and a batch size of 200. For NPE/FMPE (spectral), we use 10 modes, leading to 20
1137 parameters to learn, and a pad width of 20 for the spectral preprocessing (Appendix [S4.1](#)). For NPE
1138 (spectral) the density estimator is a NSF with 2 residual blocks with 50 hidden dimensions each, 5
1139 transforms, with RELU activations. For FMPE (spectral), we use an MLP with 5 linear layers with 64
1140 hidden dimensions to estimate the flow, with ELU activations. For FMPE (raw), we use an MLP with
1141 5 layers and 64 hidden features, with ELU activations. For all baseline methods, we used the same
1142 embedding as Redacted [\[39\]](#), which was a CNN embedding the 441-dimensional observation into a
1143 50-dimensional vector using 2 convolutional layers with kernel size 5 followed by max pooling of
1144 kernel size 2, followed by a 2-layer MLP with 50 hidden units, with RELU nonlinearities throughout.
1145 The configuration of NPE (raw) is described in Appendix [S4.3](#).

1146 For FNOPE we use 10 Fourier modes for the FNO blocks. The network is made of 5 FNO blocks
1147 with 16 channels, while the context is embedded into 8 channels. We train for a maximum of 1000
1148 epochs with an early patience of 50. We use a training batch size of 200 and a learning rate of 0.001.
1149 We do not include the data augmentation procedure for this experiment, as the discretizations of both
1150 observations and parameters was fixed to the setting of [\[39\]](#). We still include positional embedding
1151 due to the parameter and observations being discretized differently to one another: the architecture
1152 included 4 channels for positional embedding, and 4 channels for time embedding. All nonlinearities
1153 are GELUs.

1154 For the experiments on 500 gridpoints (Fig. [S5](#)) we use the same hyperparameters.

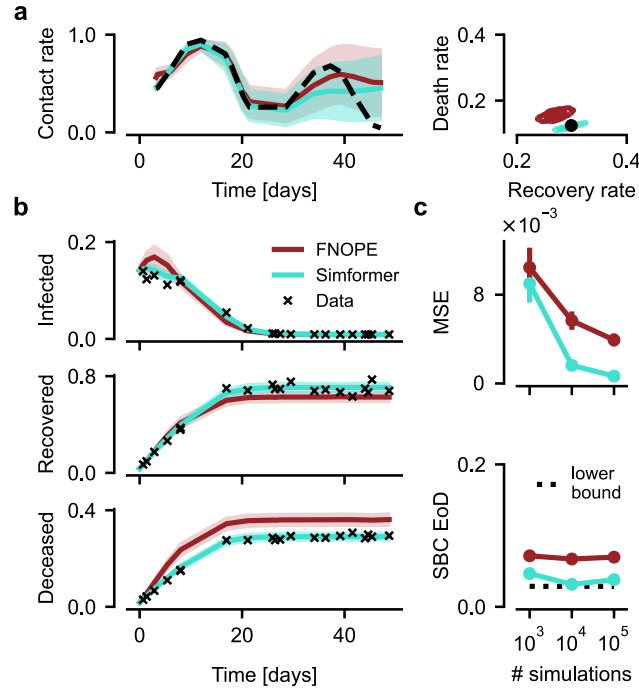


Figure S1: **SIRD model on 20 conditioning points.** (a) Posterior conditioned on 20 time points. *left:* Posterior (mean \pm std.) of the time-varying parameter and ground truth parameters (dashed). *right:* Two dimensional posterior of vector-valued parameters and ground truth parameters (dot). (b) Posterior predictive (mean \pm std.) of infected, recovered and deceased populations with observations marked. (c) *upper:* MSE of posterior predictive samples to observations. *lower:* Simulation-based calibration error of diagonal (SBC EoD). 'Lower bound' refers to the SBC EoD for uniformly sampled posterior ranks (details in Appendix S2). See Fig. 4 for the results with 40 conditioning points.

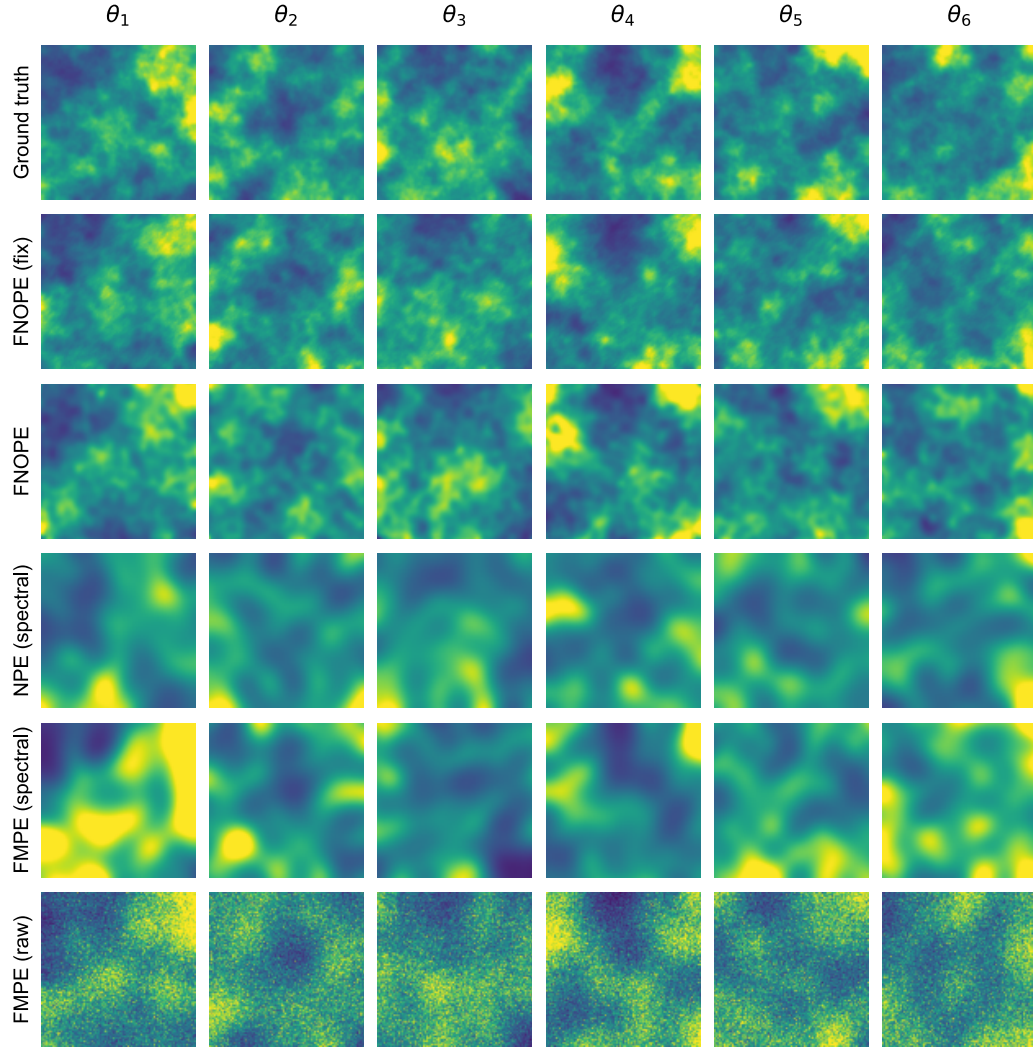


Figure S2: **Posterior samples Darcy flow experiment.** Additional posterior samples for the Darcy flow experiment, for six distinct observations simulated from ground truth parameter θ_i .

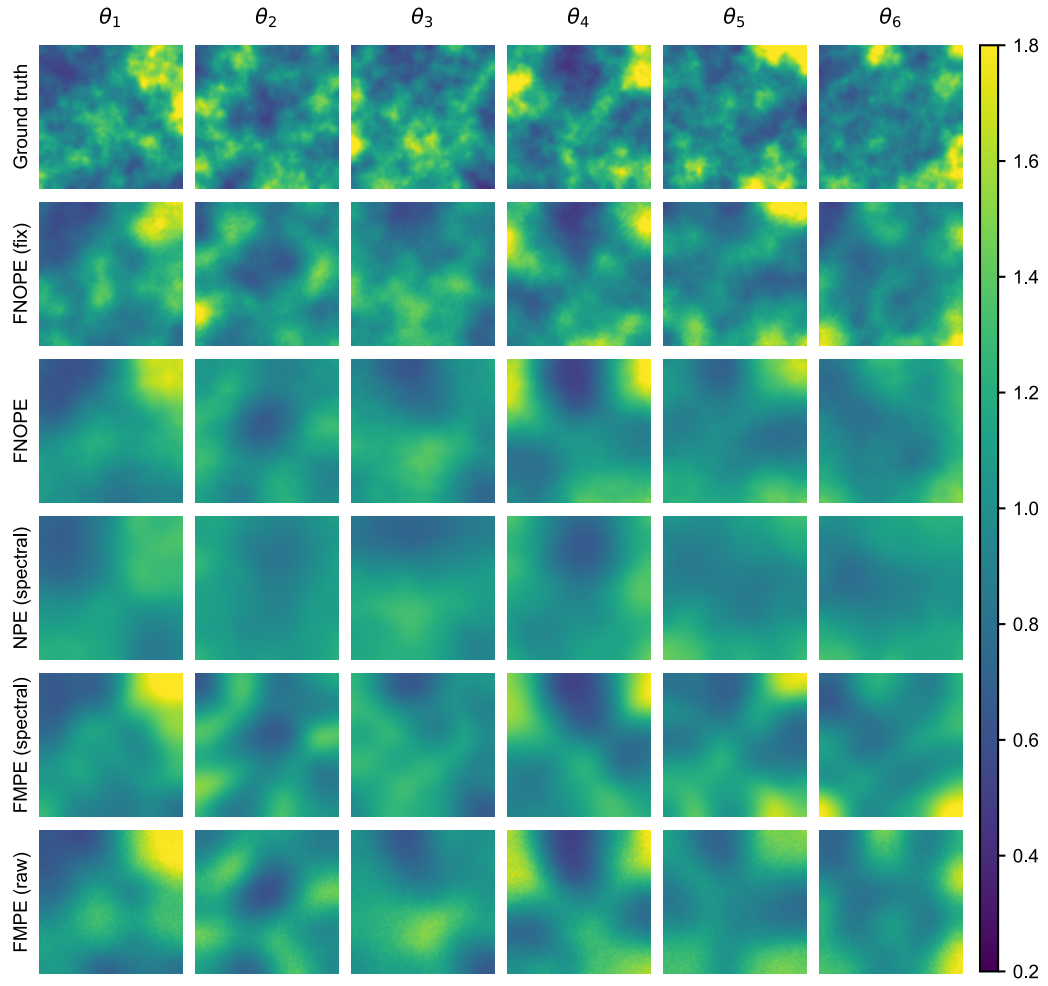


Figure S3: **Posterior means for Darcy flow experiment.** Posterior means (based on 100 samples) for the Darcy flow experiment, for six distinct observations simulated from ground truth parameter θ_i .

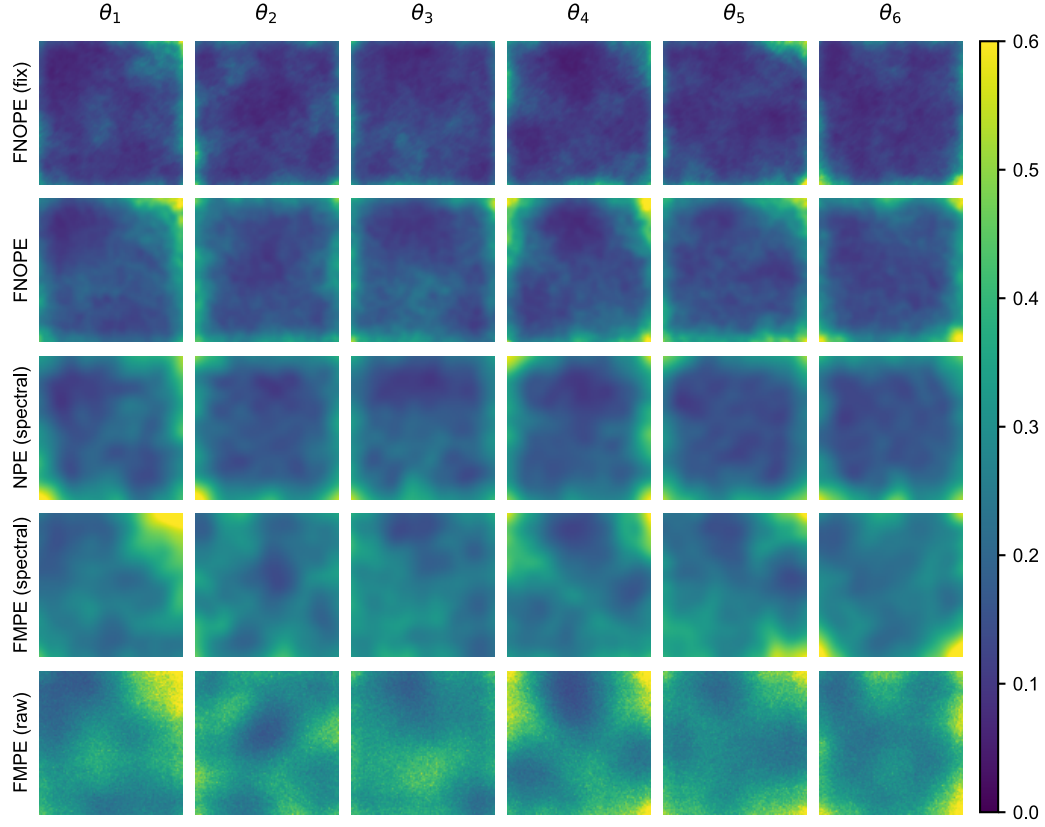


Figure S4: **Posterior standard deviation for Darcy flow experiment.** Posterior standard deviations (based on 100 samples) for the Darcy flow experiment, for six distinct observations simulated from ground truth parameter θ_i .

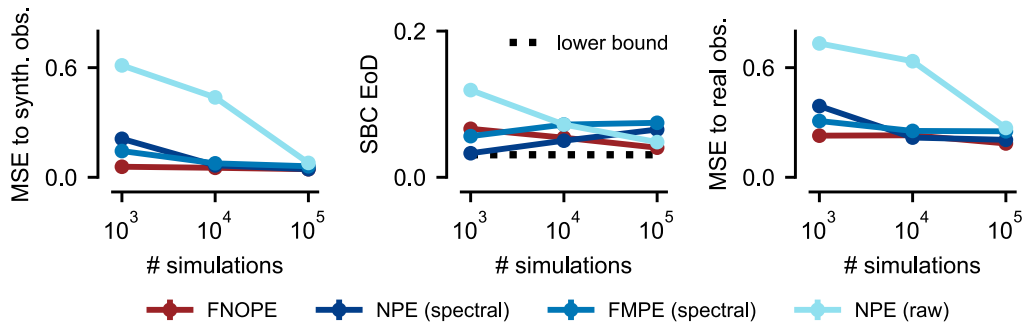


Figure S5: **Mass balance rates of ice shelves.** Inference of mass balance rates where the parameter discretization uses 500 gridpoints (observation discretization remains unchanged). Performance measures on test simulations and the real observation, where NPE (raw) refers to the method used in Redacted [39]. Results for FMPE (raw) omitted as this baseline was not able to always produce samples within the prior bounds across all test observations. See Fig. 6 for the results based on 50 grid points.