Accelerating mechanistic inference via neural density estimation

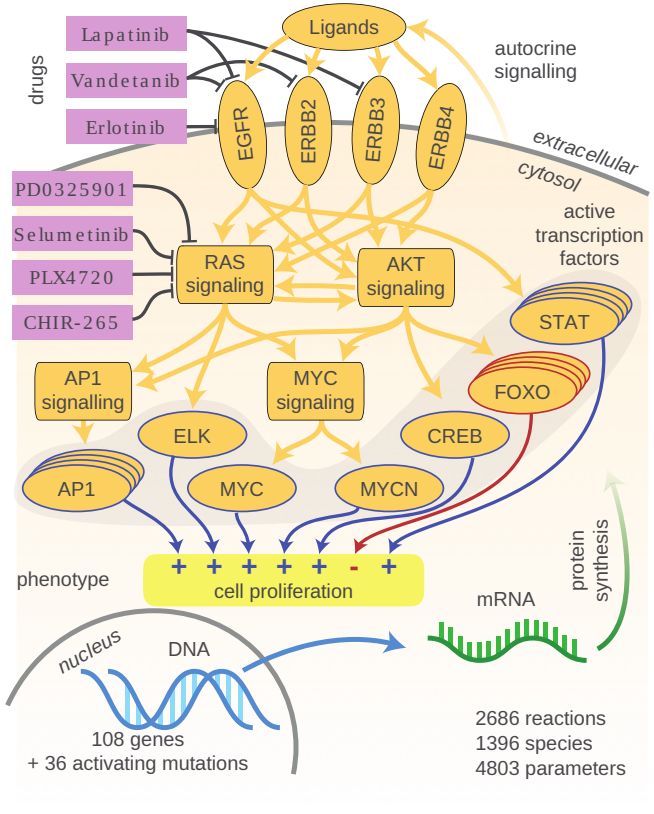
Yannik Schaelte
AstraZeneca, Computational Pathology, 2023-05-16
Journal Club

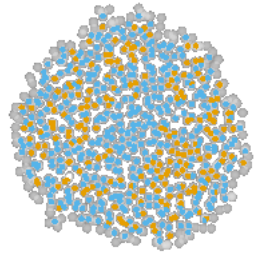
yannik-schaelte @yannik_schaelte yannikschaelte

OVERVIEW

- Quantitative systems biology
- Amortized inference
- Missing data
- Mixed-effects modeling

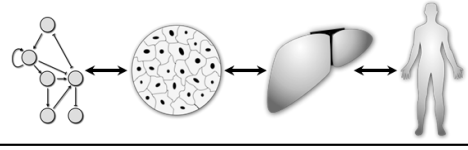
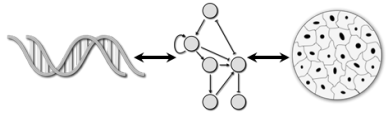
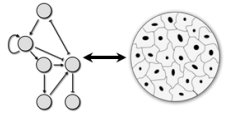
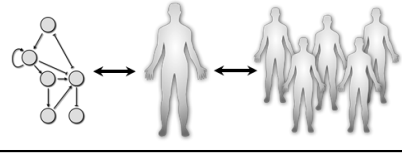
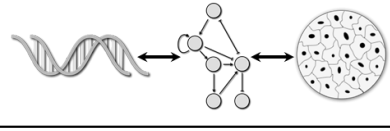
BIOLOGICAL PROCESSES ARE COMPLICATED





Fröhlich et al., Cell Systems, 2018, and Jagiella et al., Cell Systems, 2017

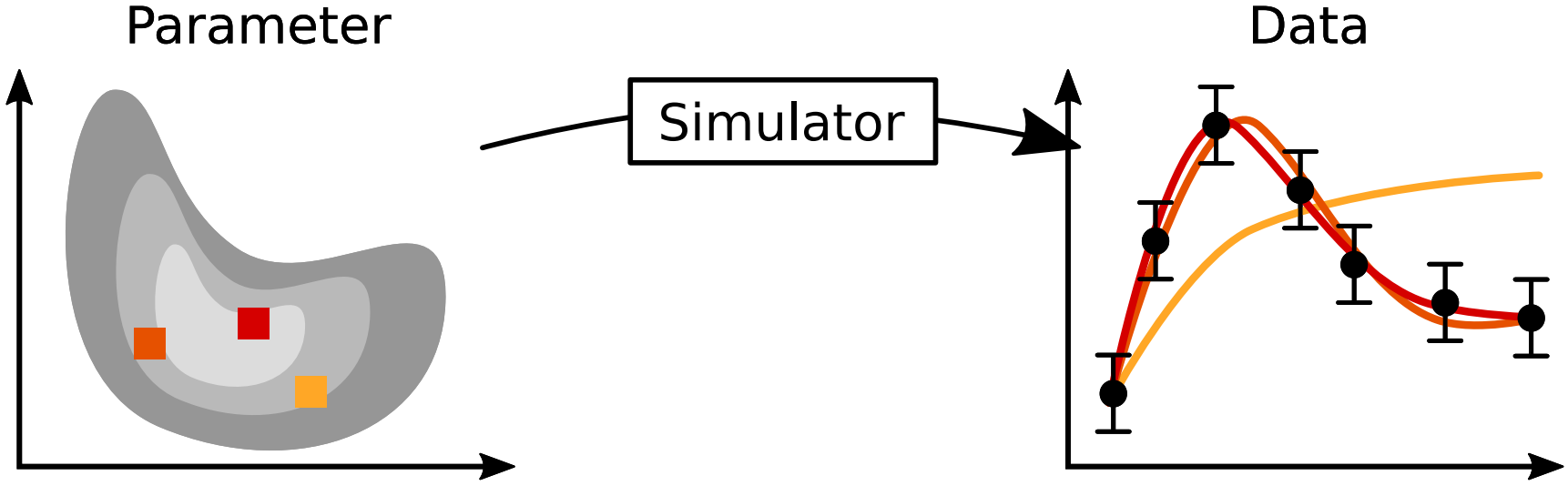
MODEL TYPES

		Boolean / Petri net models	Constraint-based models	Markov chain models	SDE models	ODE models	PDE models	Agent-based models	
whole-heart (tissue and fluid mechanics, signaling)		■				■	■		Hunter and Borg, *Nature Reviews Molecular Cell Biology*, 4(3):237-243, 2003
cancer growth (signaling, cell division and death, angiogenesis, tissue remodeling)					■		■	■	Anderson and Quaranta, *Nature Reviews Cancer*, 8(3):227-234. 2008
liver lobule (cell division and cell death, tissue mechanics)				■	■		■	■	Hoehme et al., *PNAS*, 107(23):10371-10376, 2010
glucose-insulin-glucagon regulation (blood and interstitial flow, organ uptake, signaling)						■	■	■	Schaller et al., *CPT: Pharmacometrics and Systems Pharmacology*, 2:e65, 2013
whole-cell (transcription, translation, DNA replication, metabolism, replication)		■	■	■		■		■	Karr et al., *Cell*, 150(2):389-401, 2012

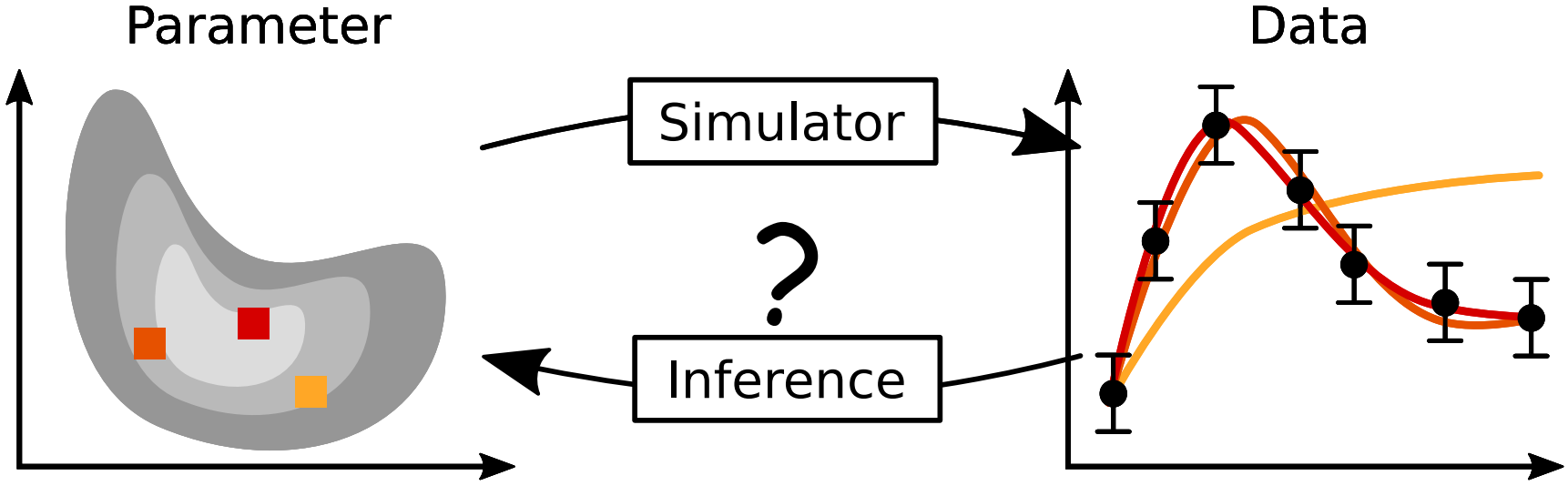
based on Hasenauer et al., Coupl. Sys. 2015

THE INVERSE PROBLEM

THE INVERSE PROBLEM

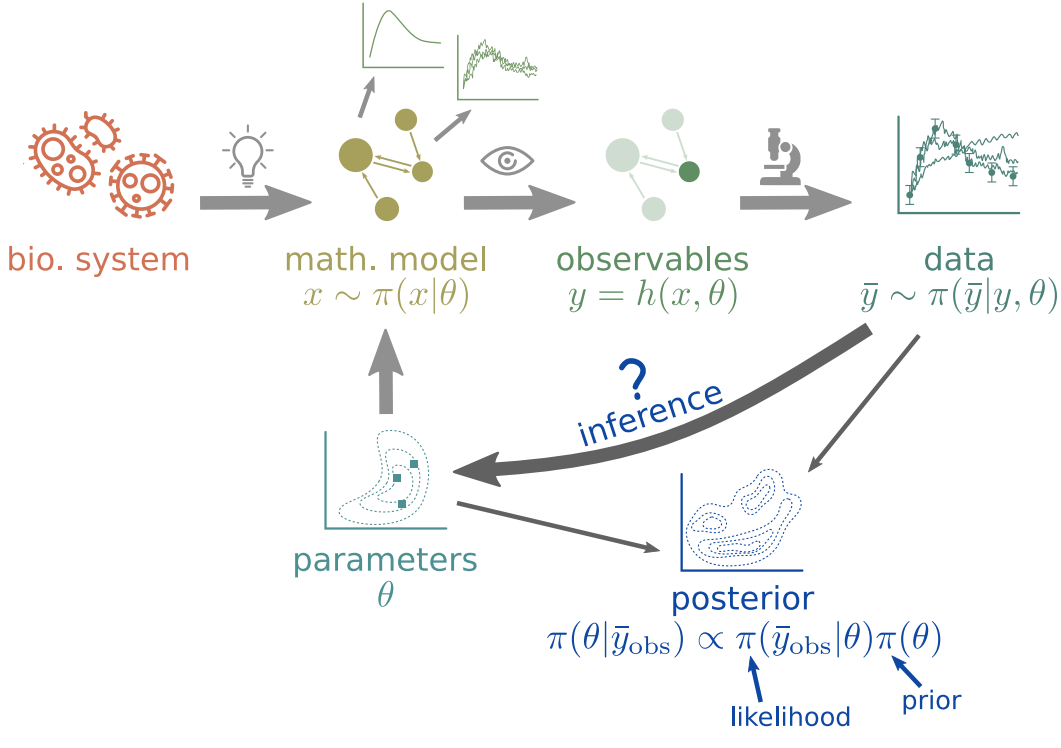


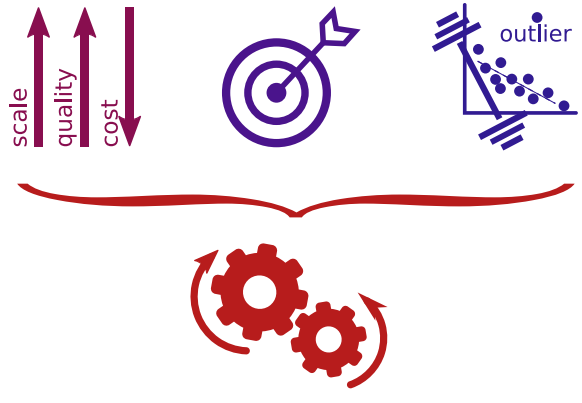
THE INVERSE PROBLEM



Efficient accurate and robust statistical inference for deterministic and stochastic models of biochemical systems

Efficient accurate and robust statistical inference for
deterministic and stochastic models of biochemical systems





SOFTWARE

ODE simulation and sensitivity analysis

parameter inference: optimization, sampling, profiles, ...

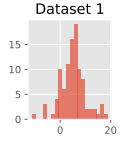
analysis of multi-scale multi-cellular models

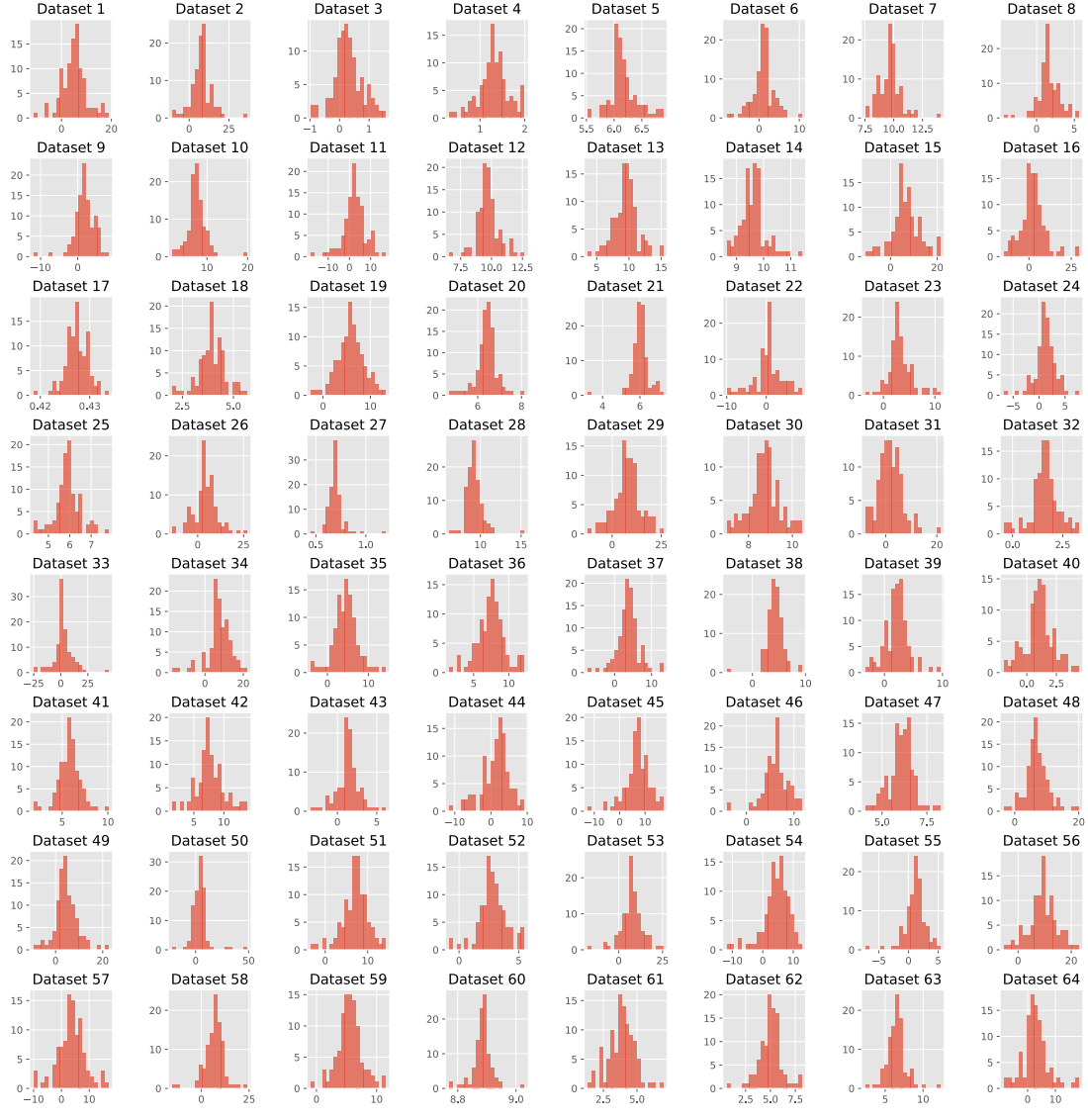
likelihood-free inference for stochastic models

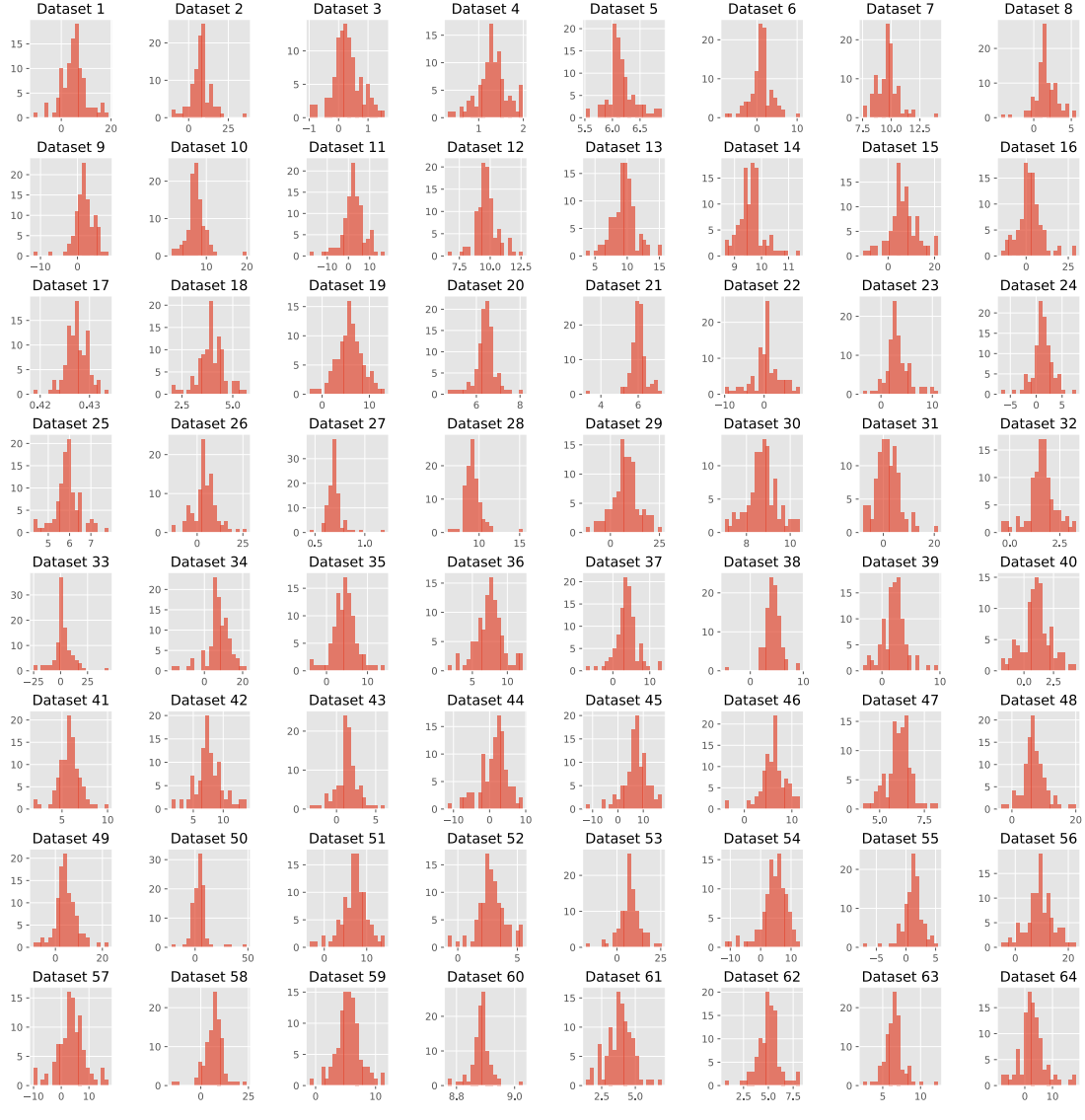
standard to specify estimation problems

definition of ODE problems in YAML and conversion to SBML

AMORTIZED INFERENCE


Dataset 1
15
10
5
0
0
20


Dataset 1
Dataset 2
Dataset 3
Dataset 4
Dataset 5
Dataset 6
Dataset 7
Dataset 8
Dataset 9
Dataset 10
Dataset 11
Dataset 12
Dataset 13
Dataset 14
Dataset 15
Dataset 16
Dataset 17
Dataset 18
Dataset 19
Dataset 20
Dataset 21
Dataset 22
Dataset 23
Dataset 24
Dataset 25
Dataset 26
Dataset 27
Dataset 28
Dataset 29
Dataset 30
Dataset 31
Dataset 32
Dataset 33
Dataset 34
Dataset 35
Dataset 36
Dataset 37
Dataset 38
Dataset 39
Dataset 40
Dataset 41
Dataset 42
Dataset 43
Dataset 44
Dataset 45
Dataset 46
Dataset 47
Dataset 48
Dataset 49
Dataset 50
Dataset 51
Dataset 52
Dataset 53
Dataset 54
Dataset 55
Dataset 56
Dataset 57
Dataset 58
Dataset 59
Dataset 60
Dataset 61
Dataset 62
Dataset 63
Dataset 64



FITTING A MODEL TO MANY DATASETS?

IN BRIEF

IN BRIEF

✗ Classical (simulation-based) parameter estimation is case-based + slow + approximate

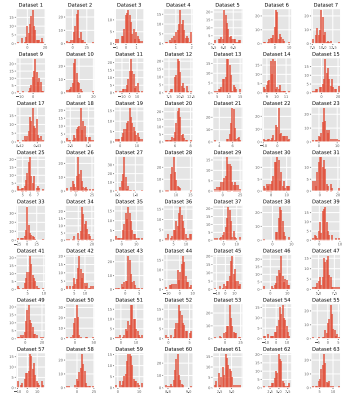
IN BRIEF

- ✗ Classical (simulation-based) parameter estimation is case-based + slow + approximate
- ✗ What if we want to fit the same model to multiple datasets?

IN BRIEF

✗ Classical (simulation-based) parameter estimation is case-based + slow + approximate

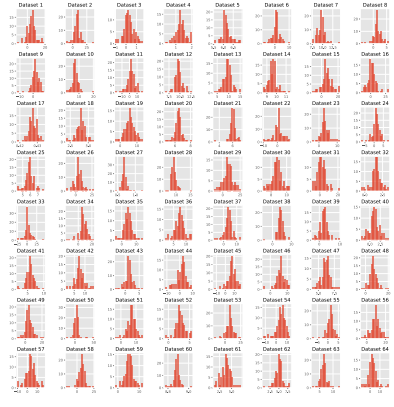
✗ What if we want to fit the same model to multiple datasets?



IN BRIEF

✗ Classical (simulation-based) parameter estimation is case-based + slow + approximate

✗ What if we want to fit the same model to multiple datasets?

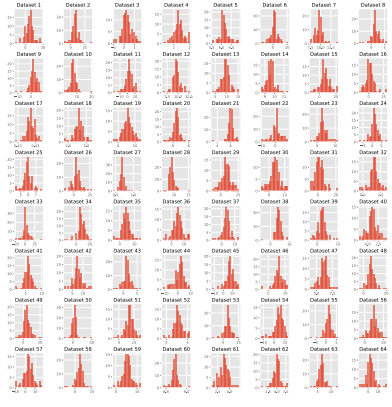


✓ Learn a global estimator for the probabilistic mapping from data to parameters

IN BRIEF

✗ Classical (simulation-based) parameter estimation is case-based + slow + approximate

✗ What if we want to fit the same model to multiple datasets?



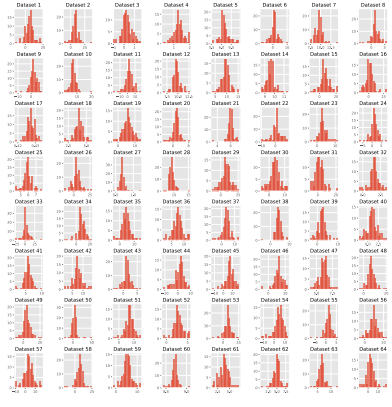
✓ Learn a global estimator for the probabilistic mapping from data to parameters

✓ Once trained, amortize inference on arbitrarily many datasets

IN BRIEF

✗ Classical (simulation-based) parameter estimation is case-based + slow + approximate

✗ What if we want to fit the same model to multiple datasets?



✓ Learn a global estimator for the probabilistic mapping from data to parameters

✓ Once trained, amortize inference on arbitrarily many datasets

✓ Embed data via summary statistics model

GENERATIVE MODELS

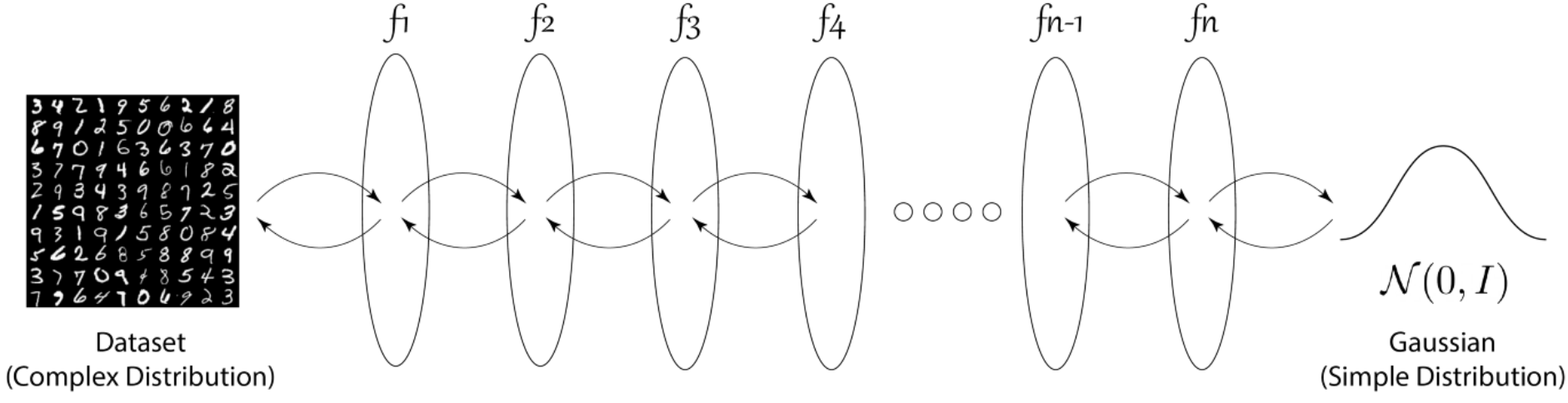
generate new data instances, $x \sim \pi(X|Y = y)$

from: Kingma et al., NeurIPS 2019

e.g.: GANs, VAEs, Flows

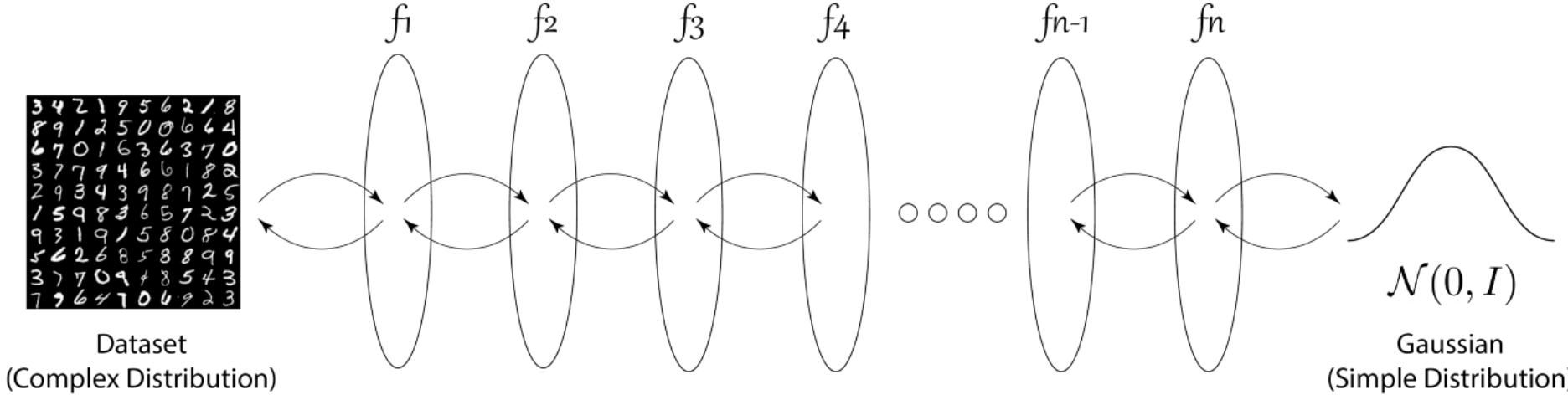
NORMALIZING FLOWS

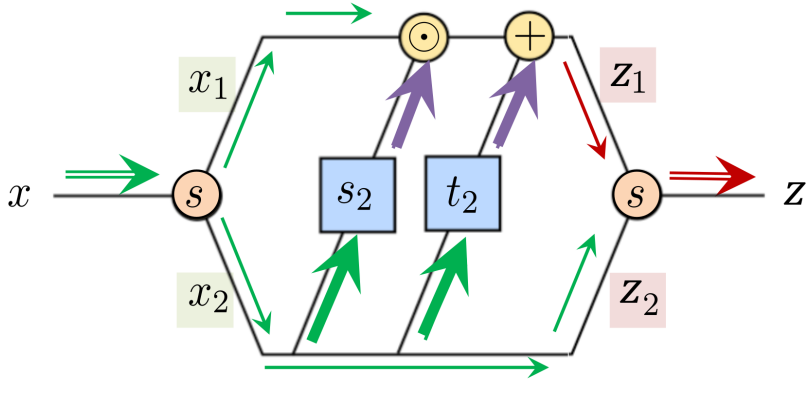
generative models based on an invertible transformation



NORMALIZING FLOWS

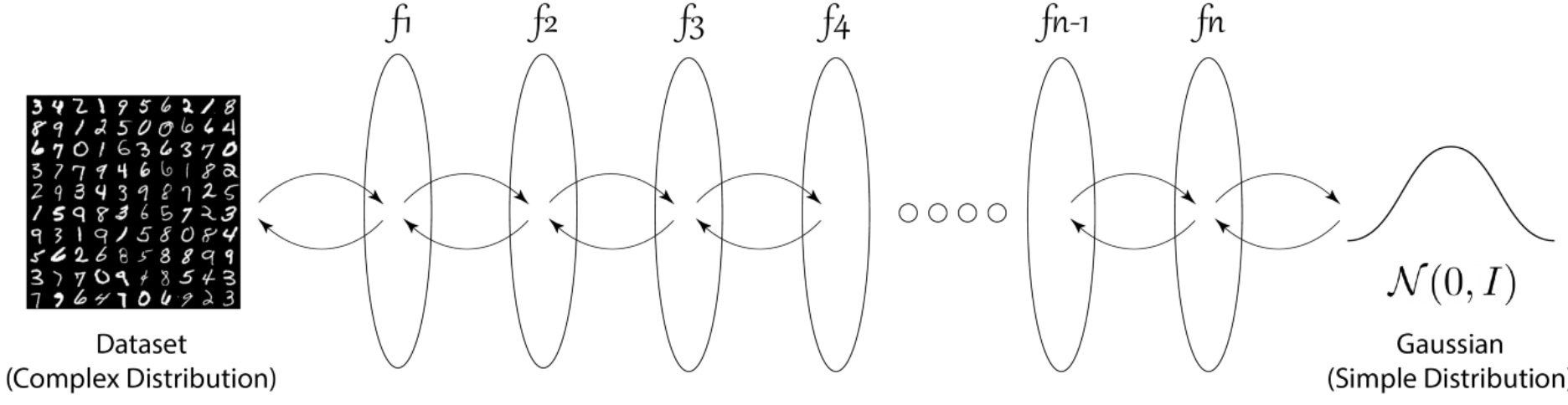
generative models based on an invertible transformation

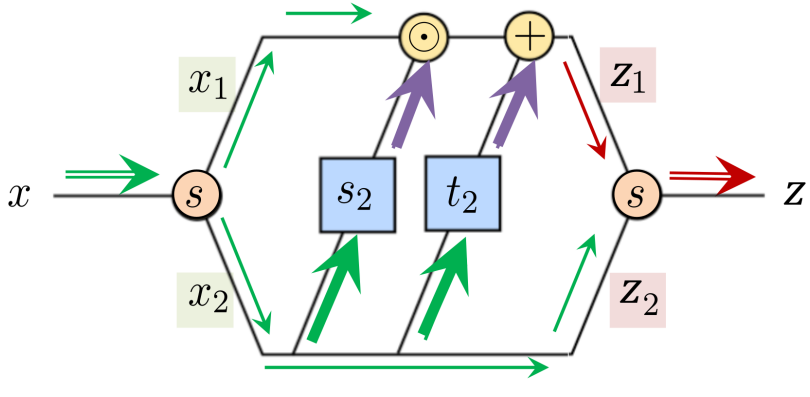


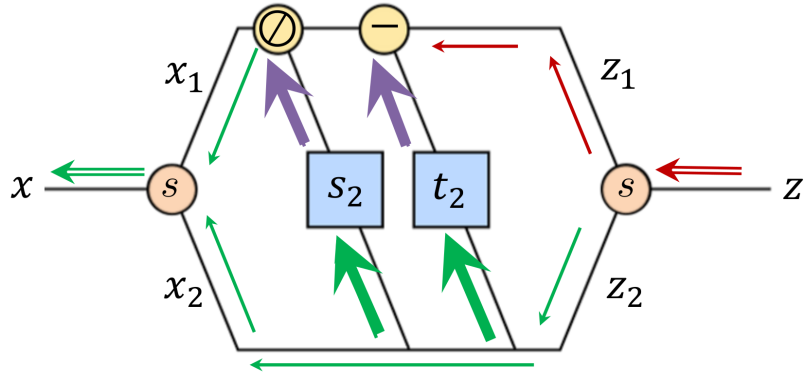


NORMALIZING FLOWS

generative models based on an invertible transformation

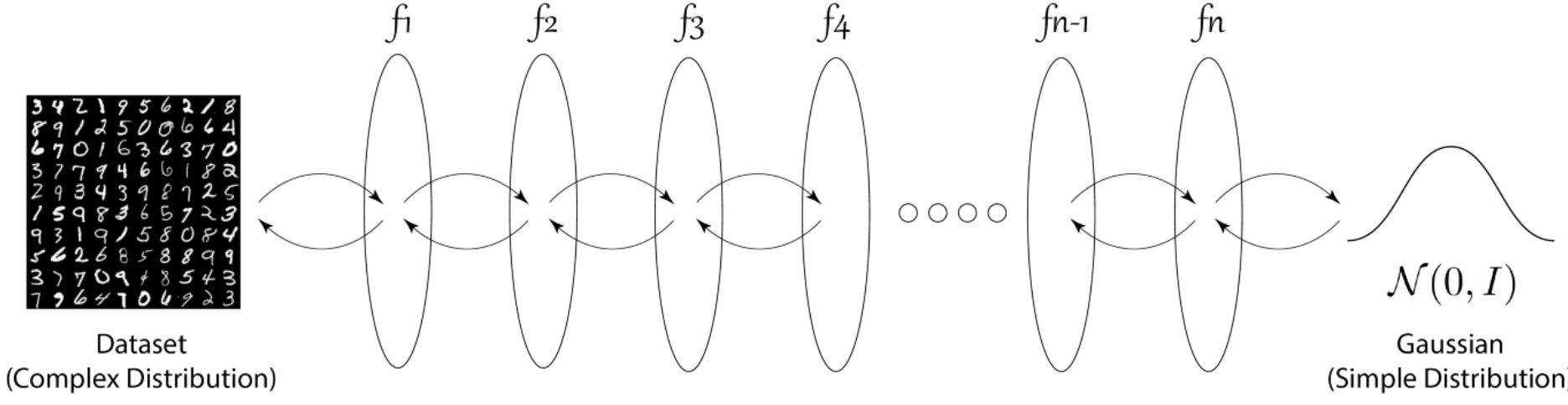


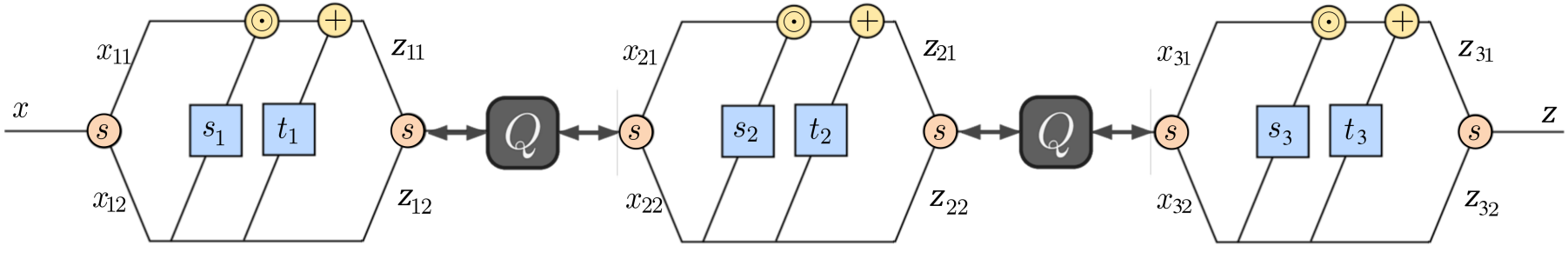




NORMALIZING FLOWS

generative models based on an invertible transformation



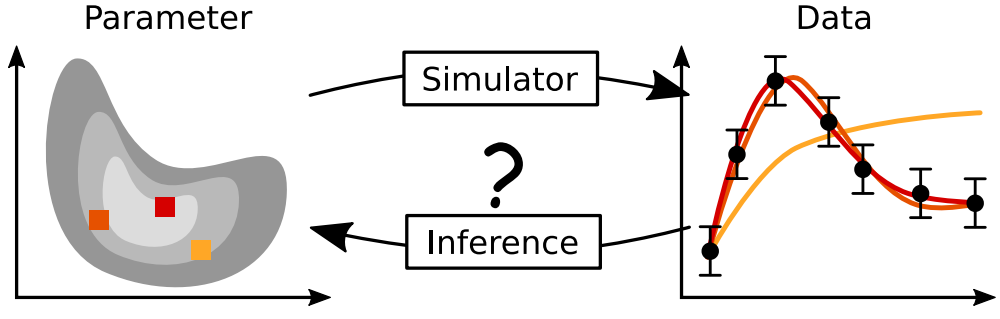


Let $z \sim \mathcal{N}(0, I)$ and $f : z \mapsto x$ bijective. Then via change of variable,s the pdf of $x = f(z)$ is given as

$$p_x(x) = p_z(f^{-1}(x)) \cdot |\det(\frac{df^{-1}}{dx}(x))|.$$

- in training, transform data points to simple distribution
- trained via negative log-likelihood
- afterwards, generate samples via $f^{-1}(z)$ with $z \sim \mathcal{N}(0, I)$

POSTERIOR LEARNING



- parameters θ
- observations x
- forward model $x \sim p(x|\theta) \Leftrightarrow x = g(\theta, \xi)$ with $\xi \sim p(\xi)$
- Bayesian posterior $p(\theta|x) \propto p(x|\theta)p(\theta)$

- aim: find a NF $p_\phi(\theta|x) \approx p(\theta|x) \, \forall \theta, x$

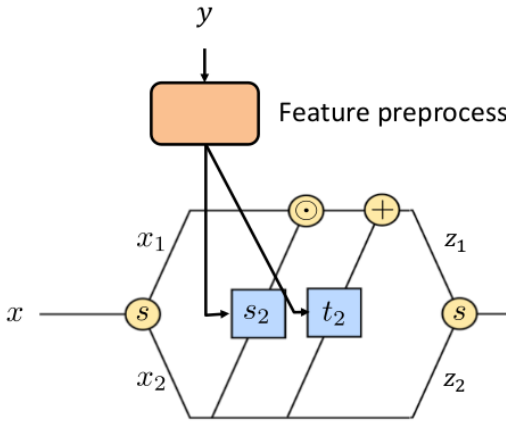
THE METHOD

Parameterize p_ϕ in terms of a cINN given via a bijective

$$f_\phi : \mathbb{R}^D \to \mathbb{R}^D, \quad \theta \mapsto z,$$

which implements a normalizing flow between θ and a Gaussian latent variable z,

$$\theta \sim p_\phi(\theta|x) \Leftrightarrow \phi = f_\phi^{-1}(z; x) \quad \text{with} \quad z \sim \mathcal{N}_D(z|0, I).$$



Seek neural network parameters $\hat{\phi}$ that minimize the KL divergence between true and approximate posterior $\forall x$

THE METHOD

$$\hat{\phi} = \arg\min_{\phi} \mathbb{E}_{p(x)}[\mathrm{KL}(p(\theta|x) \,||\, p_\phi(\theta|x))]$$

$$= \arg\max_{\phi} \iint p(x,\theta) \log p_\phi(\theta|x) dx d\theta$$

$$= \arg\max_{\phi} \iint p(x,\theta)(\log p(f_\phi(\theta; x)) + \log |\det \boldsymbol{J}_{f_\phi}|) dx d\theta$$

Approximate via Monte-Carlo sample:

$$\hat{\phi} = \arg\min_{\phi} \frac{1}{M} \sum_{m=1}^{M} (-\log p(f_\phi(\theta^{(m)}; x^{(m)})) - \log |\det \boldsymbol{J}_{f_\phi}^{(m)}|$$

$$= \arg\min_{\phi} \frac{1}{M} \sum_{m=1}^{M} \left(\frac{|f_\phi(\theta^{(m)}; x^{(m)})|_2^2}{2} - \log |\det \boldsymbol{J}_{f_\phi}^{(m)}| \right)$$

LEARN SUMMARY STATISTICS

If data $x_{1:N}$ are high-dimensional: Jointly learn a summary network $\tilde{x} = h_\psi(x_{1:N})$, giving the objective

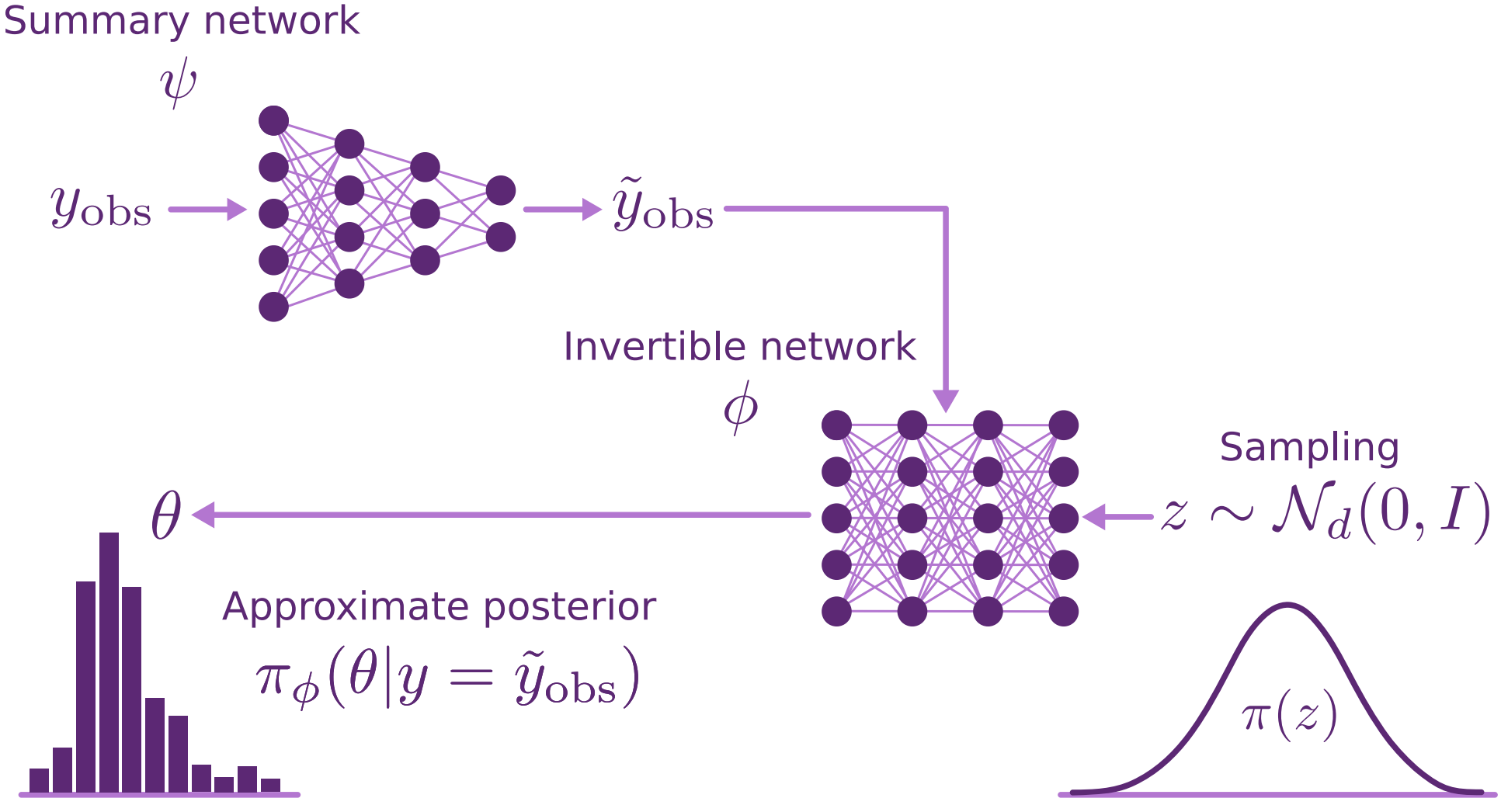
$$\hat{\phi}, \hat{\psi} = \arg\max_{\phi,\psi)} \mathbb{E}_{p(x,\theta,N)}[\log p_\phi(\theta | h_\psi(x_{1:N})]$$

with Monte-Carlo estimate

$$\hat{\phi}, \hat{\psi} = \arg\min_{\phi,\psi} \frac{1}{M} \sum_{m=1}^{M} \left(\frac{|f_\phi(\theta^{(m)}; h_\psi(x_{1:N}^{(m)})|_2^2}{2} - \log |\det(\boldsymbol{J}_{f_\phi}^{(m)})| \right)$$

AMORTIZED INFERENCE VIA INNS

you have to solve many similar problems? amortize the solution!



based on: Radev et al., 2021

TRAINING AND INFERENCE

Training phase:

- create plenty of synthetic data $(y_i, \theta_i) \sim \pi(y, \theta)$
- train a cINN in forward mode

Inference phase:

- sample many latent $z_i \sim \pi(z)$
- run cINN backwards, $\theta_i = g(z_i; y_{\text{obs}}) \sim \pi(\theta | y_{\text{obs}})$

✓ fast + accurate amortized simulation-based Bayesian inference

$$\frac{\partial y}{\partial x}$$

mechanistic modeling

$$\frac{\partial y}{\partial x}$$

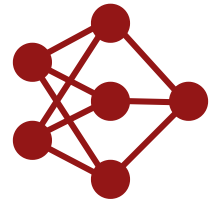
mechanistic modeling

- theory-driven
- interpretation
- testability

$$\frac{\partial y}{\partial x}$$

mechanistic modeling

- theory-driven
- interpretation
- testability

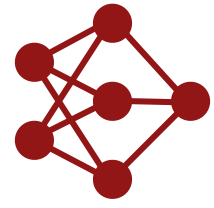


machine learning

$$\frac{\partial y}{\partial x}$$

mechanistic modeling

- theory-driven
- interpretation
- testability



machine learning

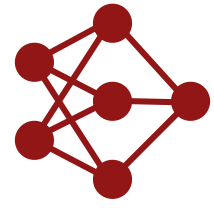
- data-driven
- large data
- automation

$$\frac{\partial y}{\partial x}$$

mechanistic modeling

- theory-driven
- interpretation
- testability

+



machine learning

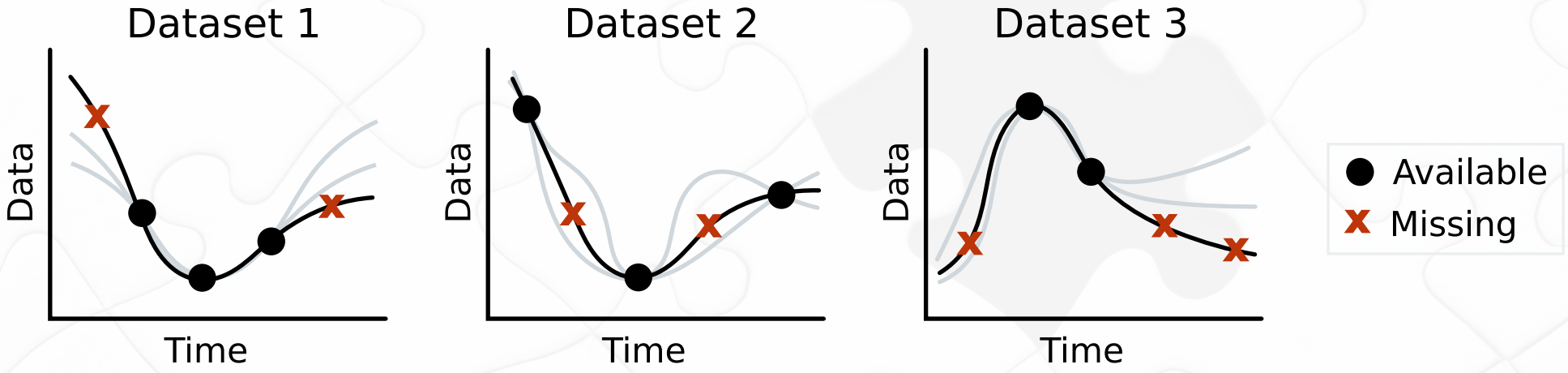
- data-driven
- large data
- automation

sciML

model-based data-efficient ML

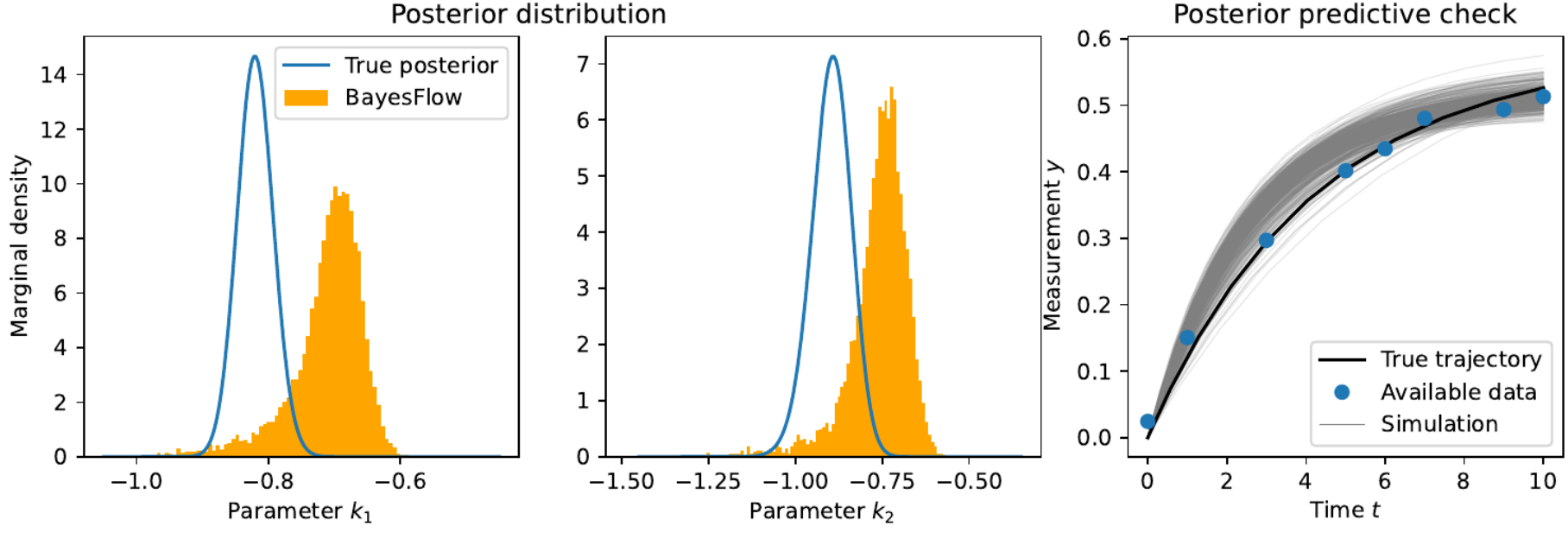
MISSING DATA

HOW TO HANDLE MISSING DATA IN AMORTIZED INFERENCE?



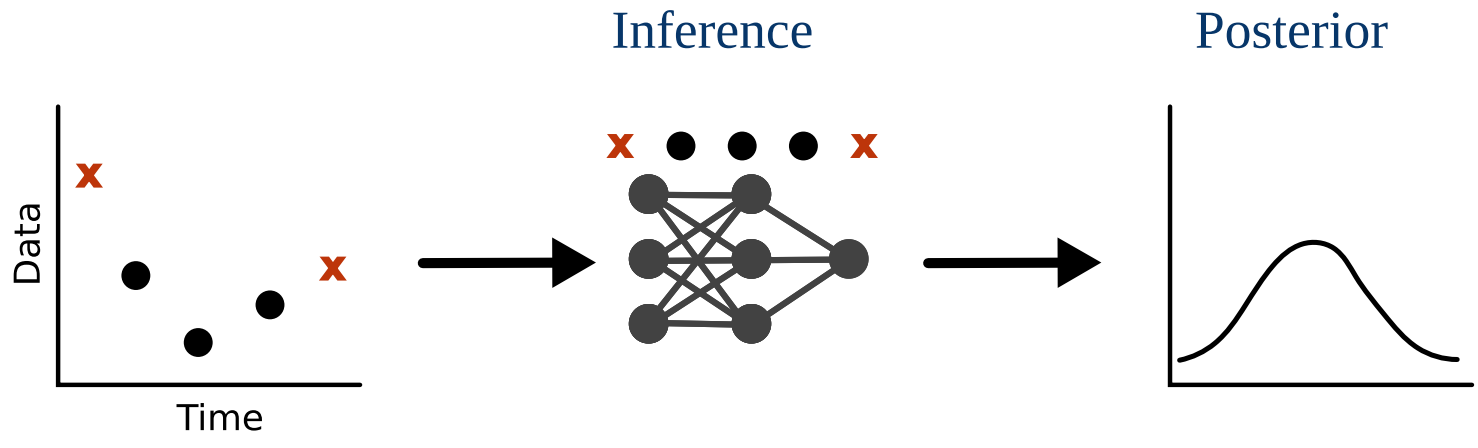
HOW TO HANDLE MISSING DATA IN AMORTIZED INFERENCE?

PROBLEM: INN CANNOT INTERPRET THE DATA

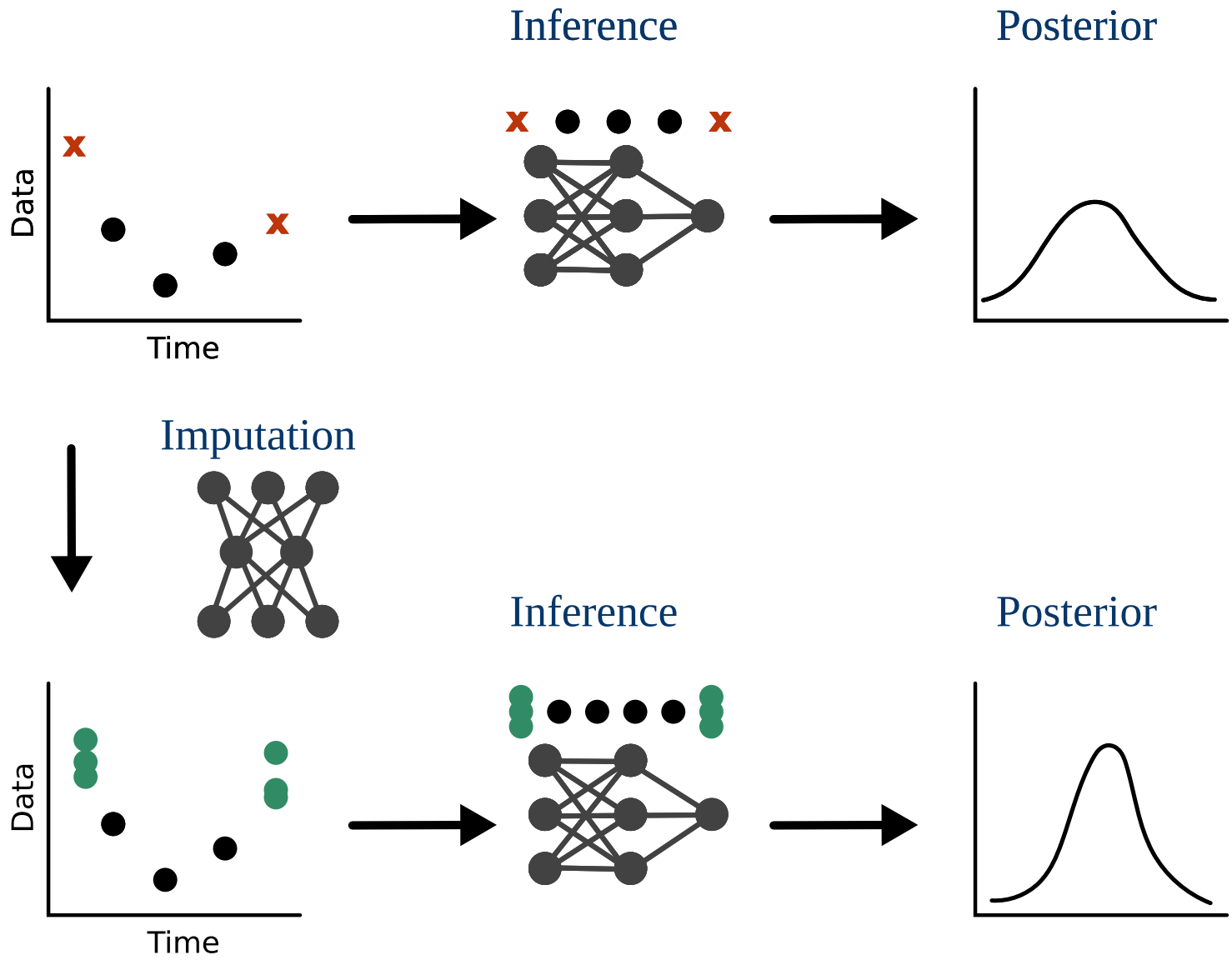


(besides iid + time-series data of heterogeneous length)

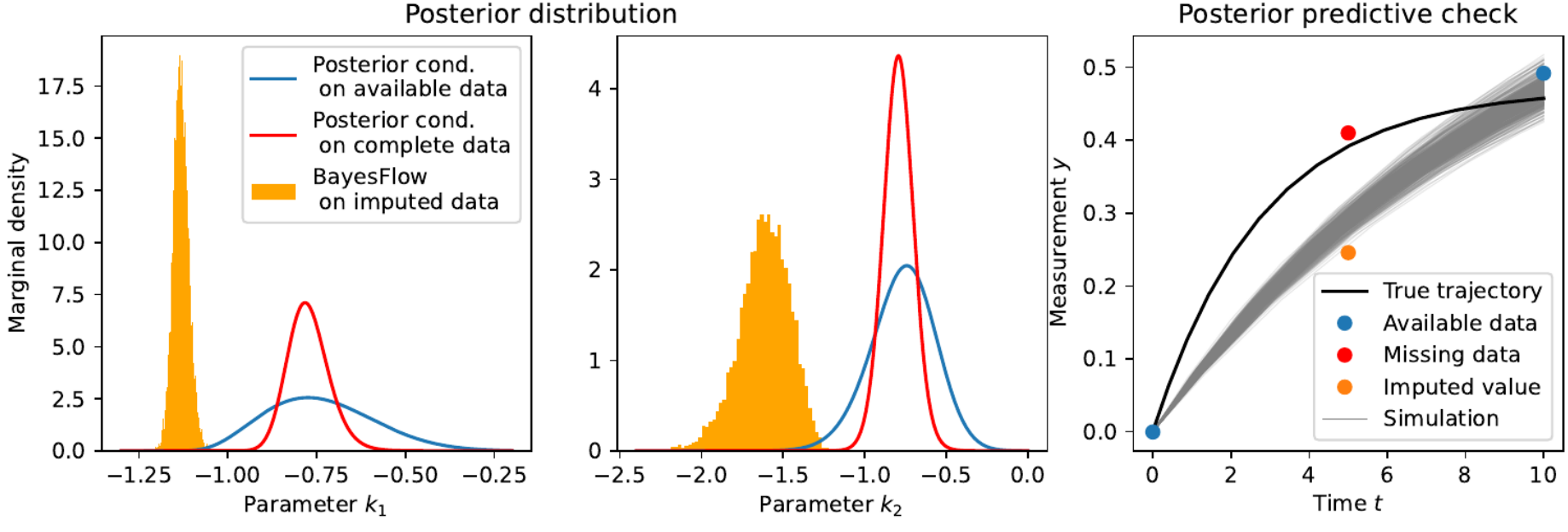
CAN WE JUST IMPUTE MISSING VALUES?



CAN WE JUST IMPUTE MISSING VALUES?



INAPPROPRIATE IMPUTATION CAN LEAD TO BIASED RESULTS



THERE ARE NO FREE DATA

Imputation means that instead of working with available data x, we try to reconstruct the full data $\bar{x}$, and estimate parameter probabilities $\pi(\theta|\bar{x})$ instead of $\pi(\theta|x)$. However, the true full data are unknown, therefore we need to take uncertainty in $\bar{x}$ into account, considering a full distribution of values $\pi(\bar{x}|x)$.

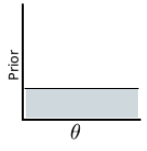
We must either make up a distribution (introducing a bias), or use a faithful approximation $p(\bar{x}|x) = \pi(\bar{x}|x)$ where $\pi(\bar{x}|x)\pi(x) = \pi(\bar{x}, x)$.

However, if we integrate out over all possible realizations of full data, we obtain $\int \pi(\theta|\bar{x})\pi(\bar{x}|x)d\bar{x} = \pi(\theta|x)$ (or similarly $\pi(\theta|x, \tau)$).

TLDR: With correct uncertainty quantification (which is hard), we just recover the same posterior.

ENCODE MISSING DATA

Training phase



Sample from prior
$\theta \sim \pi(\theta)$

Sample simulated full data
$x \sim \pi(x|\theta)$

Example

$x = \begin{bmatrix} 1.8 & 0.9 & 0.1 & 3.3 & 4.1 \end{bmatrix}$

Sample missing entries
$\tau \sim \pi(\tau)$

$\tau = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 \end{bmatrix}$

Simulated incomplete data
$x_{\text{avai}} = x \odot \tau + \text{NA} \cdot (1 - \tau)$

$x_{\text{avai}} = \begin{bmatrix} 1.8 & \text{NA} & 0.1 & 3.3 & \text{NA} \end{bmatrix}$

Encode missing data
$x_{\text{aug}} = x_{\text{aug}}(x_{\text{avai}})$

$x_{\text{aug}} = \begin{bmatrix} 1.8 & -1.0 & 0.1 & 3.3 & -1.0 \\ 1 & 0 & 1 & 1 & 0 \end{bmatrix}$

Train BayesFlow networks

loss, gradient

Missingness encoding

Insert *c*

$x_{\text{aug}} = \begin{bmatrix} 1.8 & -1.0 & 0.1 & 3.3 & -1.0 \end{bmatrix}$

Augment by 0/1

$x_{\text{aug}} = \begin{bmatrix} 1.8 & -1.0 & 0.1 & 3.3 & -1.0 \\ 1 & 0 & 1 & 1 & 0 \end{bmatrix}$

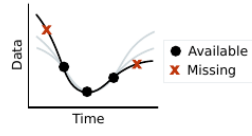
Time labels

$x_{\text{aug}} = \begin{bmatrix} 1.8 & 0.1 & 3.3 \\ 1 & 3 & 4 \end{bmatrix}$

Inference phase

Experiment

Example



Incomplete data
$x^{\text{obs}}_{\text{avai}}$

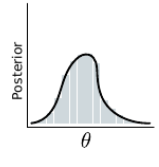
$x^{\text{obs}}_{\text{avai}} = \begin{bmatrix} \text{NA} & 3.9 & 2.5 & 3.4 & \text{NA} \end{bmatrix}$

Encode missing data
$x^{\text{obs}}_{\text{aug}} = x_{\text{aug}}(x^{\text{obs}}_{\text{avai}})$

$x^{\text{obs}}_{\text{aug}} = \begin{bmatrix} -1.0 & 3.9 & 2.5 & 3.4 & -1.0 \\ 0 & 1 & 1 & 1 & 0 \end{bmatrix}$

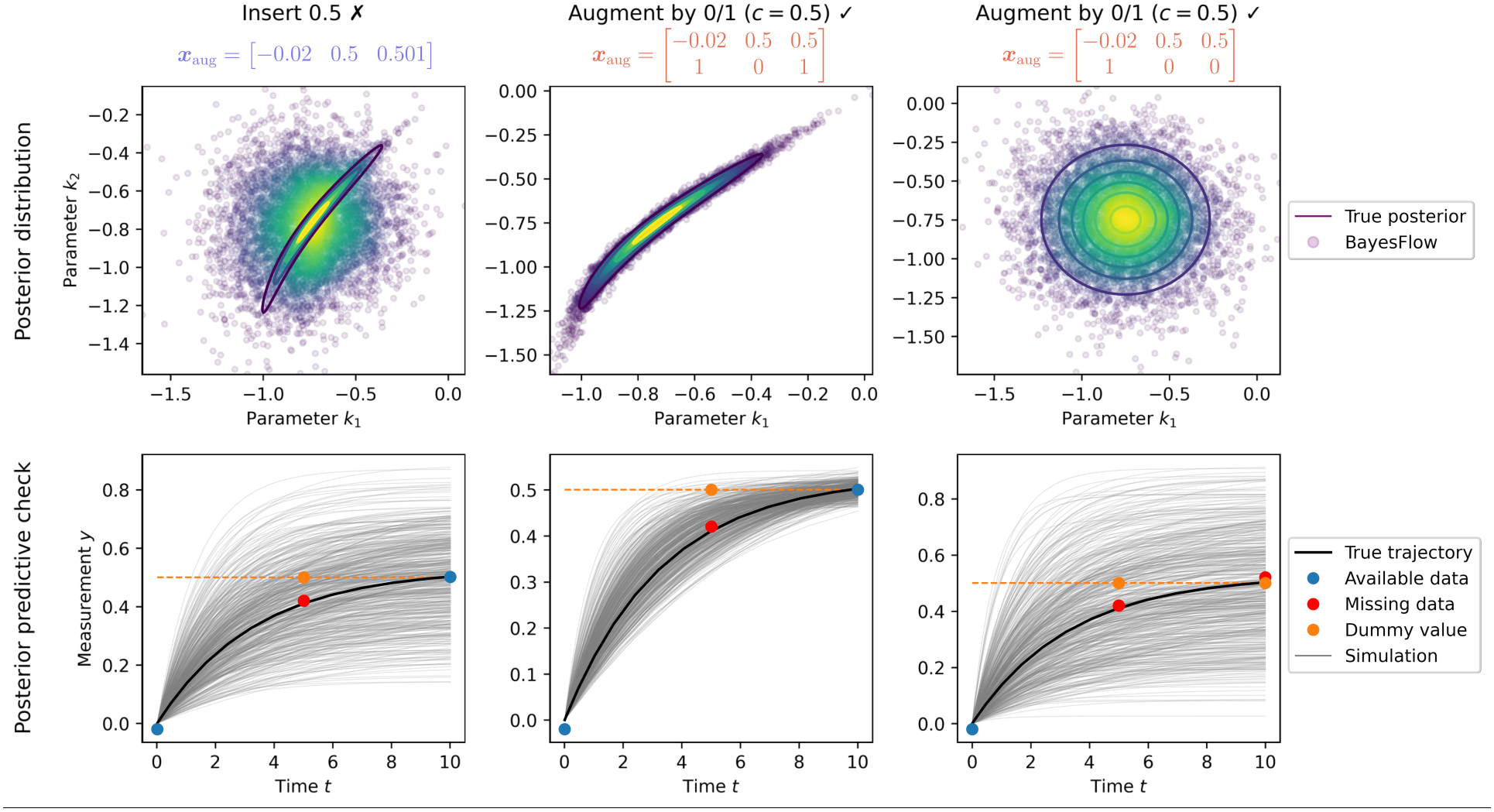
Sample from BayesFlow
$\theta \sim \pi(\theta|x^{\text{obs}}_{\text{avai}})$

Posterior approximation
$\{\theta_l\}_{l=1}^{L}$

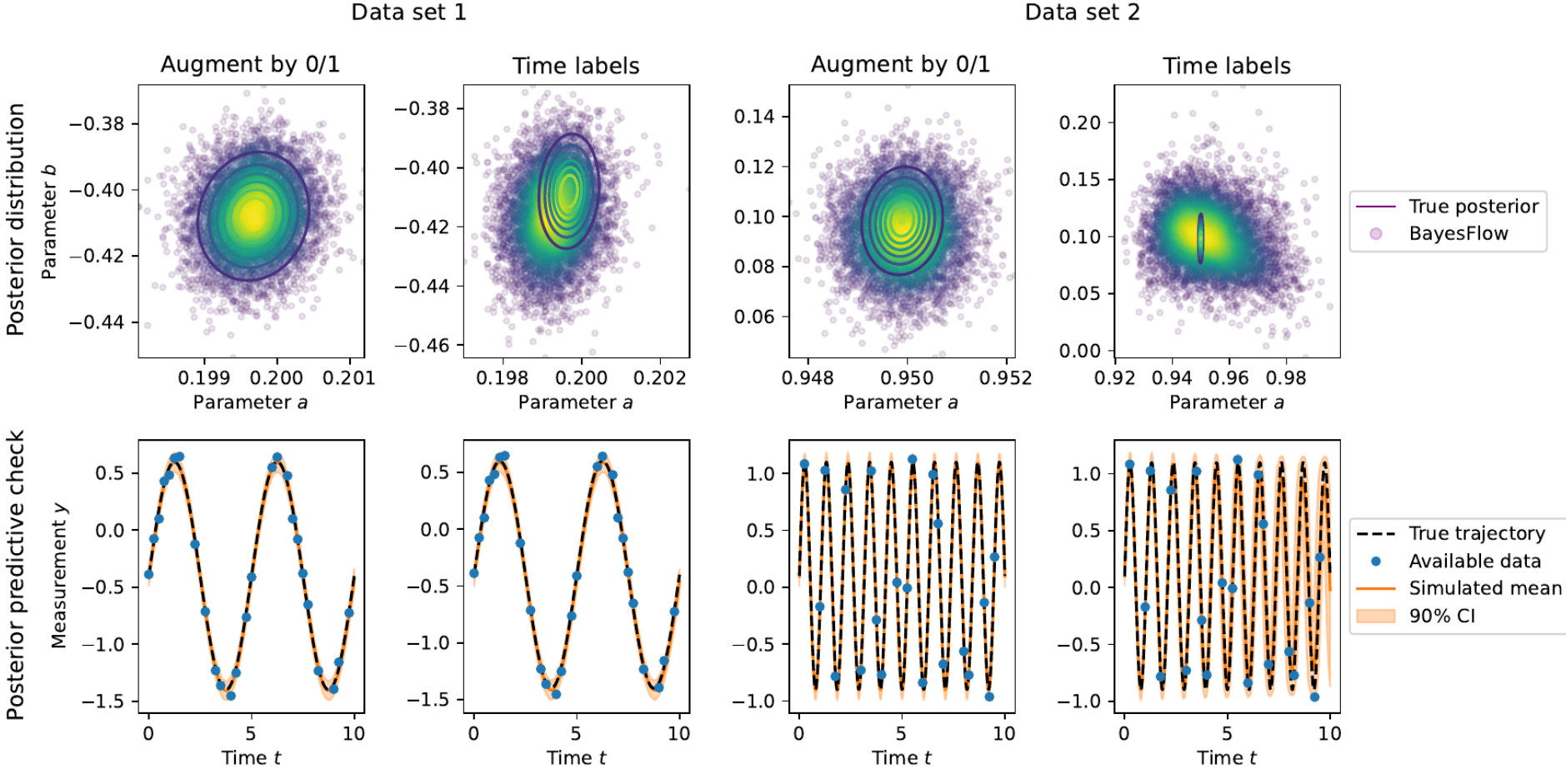


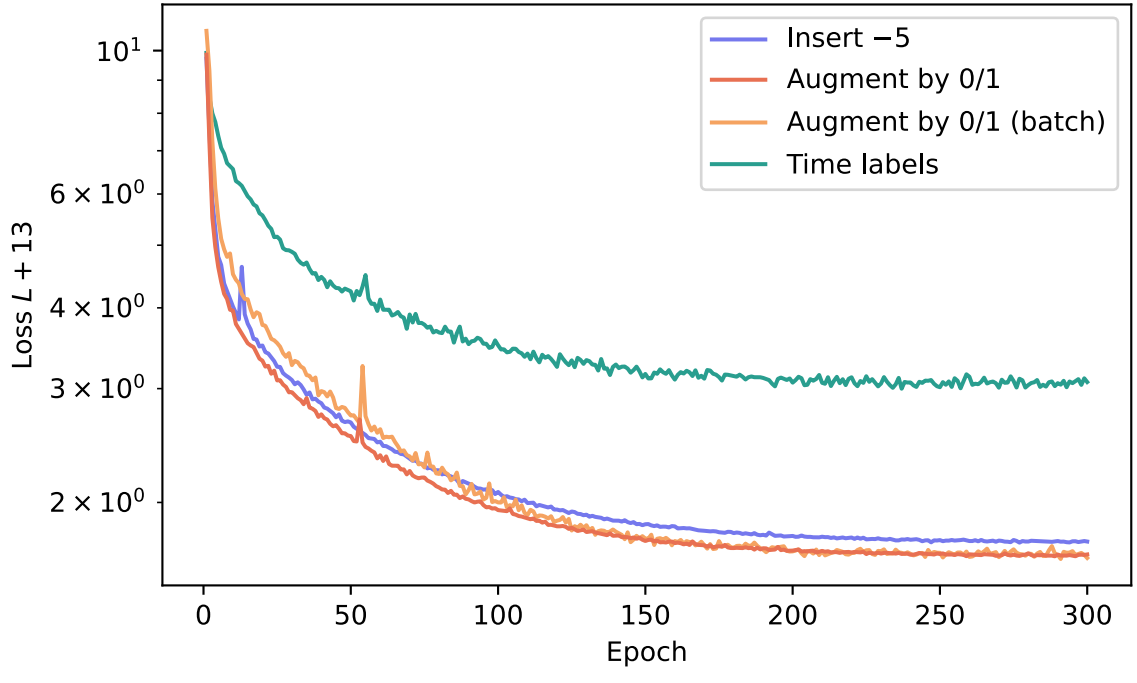
ALL APPROACHES PERFORM WELL ON SIMPLE TEST PROBLEM

BINARY INDICATOR AUGMENTATION MORE ROBUST FOR AMBIGUOUS FILL-IN VALUES

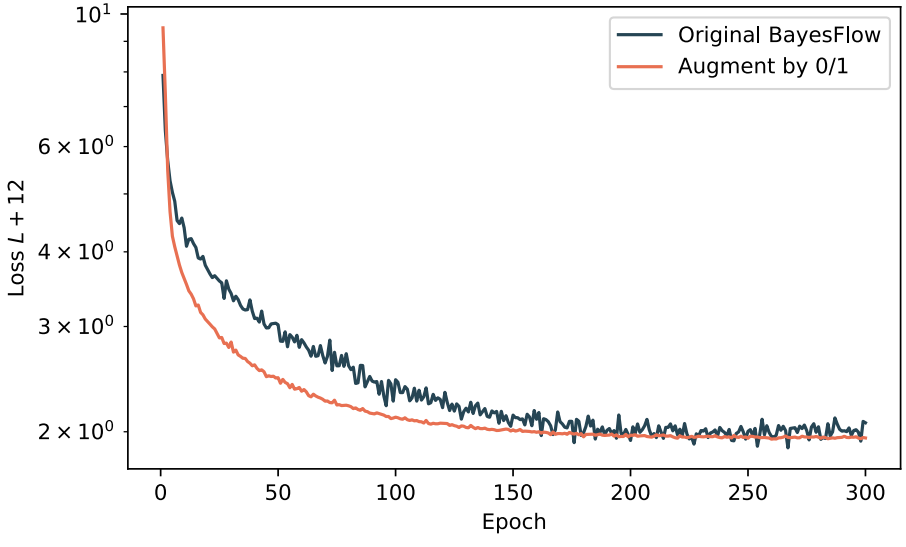


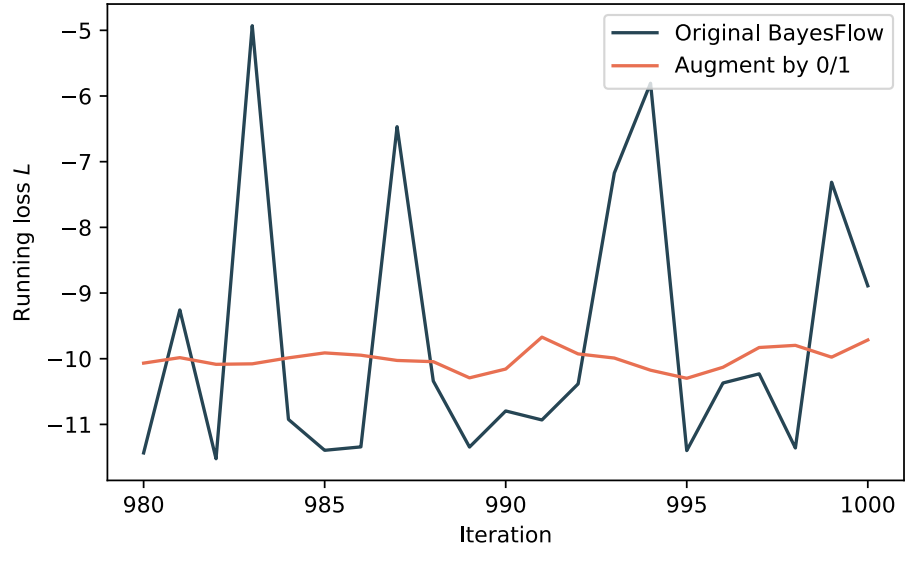
POSITIONAL ENCODING NOT ROBUST ON OSCILLATORY DATA




Insert −5
Augment by 0/1
Augment by 0/1 (batch)
Time labels
10^1
6×10^0
4×10^0
3×10^0
2×10^0
Loss $L + 13$
0
50
100
150
200
250
300
Epoch

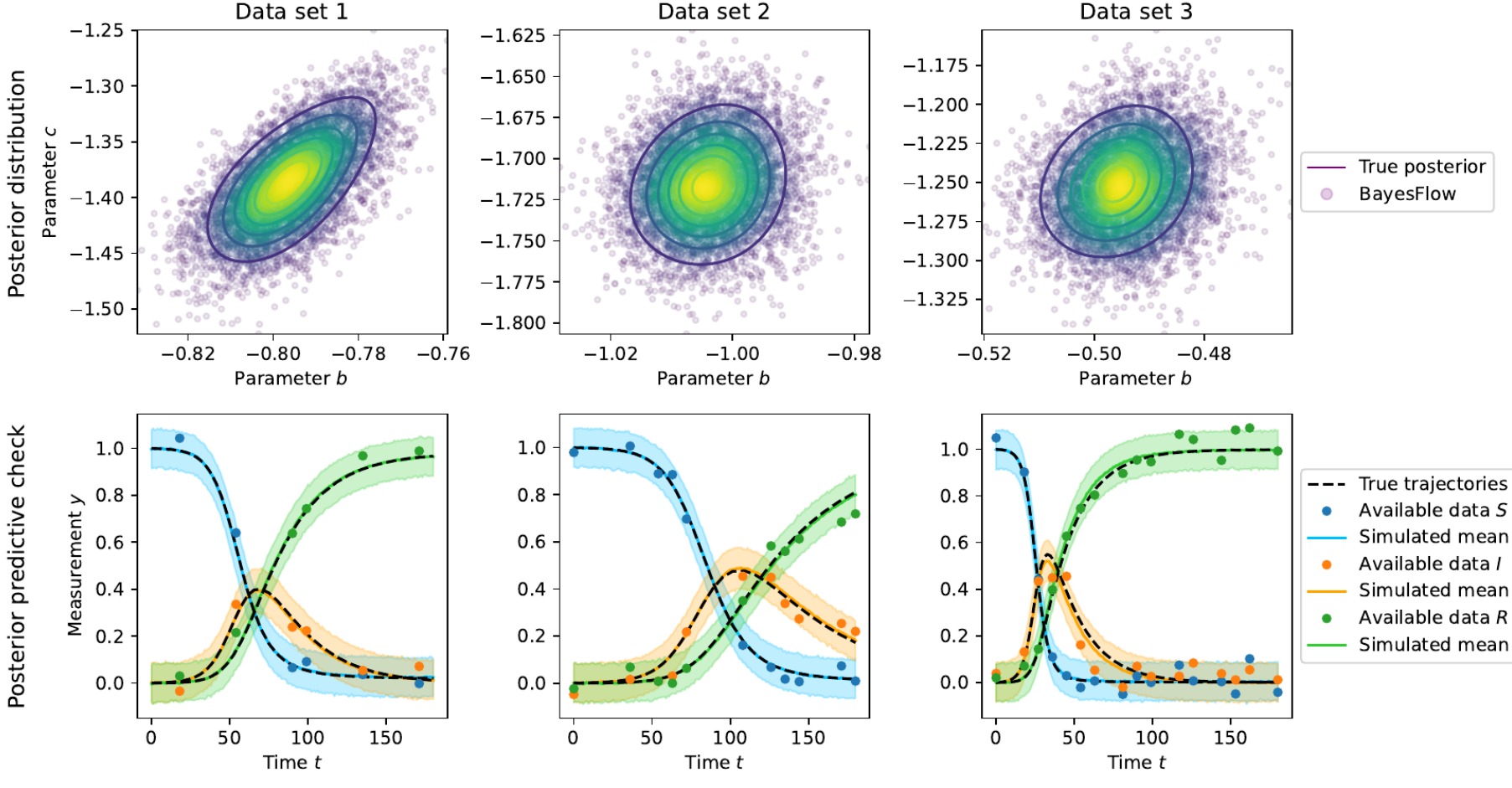
VARIABLE DATASET SIZE AS A SPECIAL CASE OF MISSING DATA



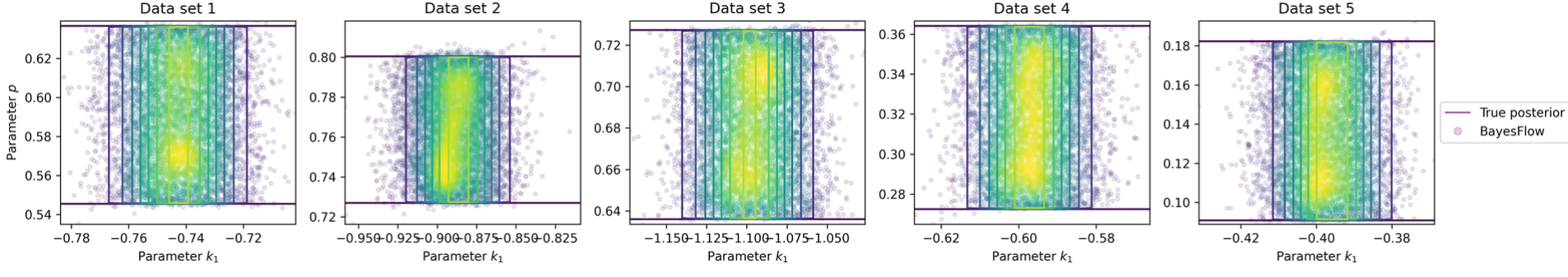


Augment by 0/1 improves performance due to better cost function approximation with individual-specific missingness

SCALES TO COMPLEX DYNAMICS



ABLE TO UNRAVEL PARAMETER-DEPENDENT MISSINGNESS



MIXED-EFFECTS MODELS

MIXED-EFFECTS MODELING

MIXED-EFFECTS MODELING

dynamical model: $\dot{x} = f(x, \theta)$

observables: $y = h(x, \theta) + \varepsilon$

parameters: $\theta = A\alpha + B\beta, \quad \beta \sim \mathrm{N}(0, \Sigma)$

PROBLEM

- **estimate parameters:** maximize over α and Σ the likelihood of data y_{obs}, marginalized over random effects β,

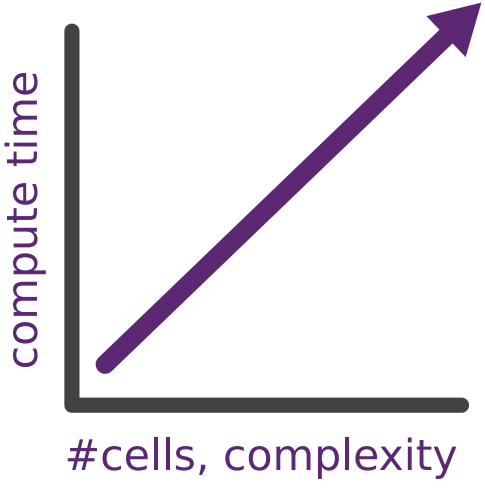
$$\pi(y_{\text{obs}}|\alpha, \Sigma) = \prod_i \int \pi(y_i|\theta)\pi(\theta|\alpha, \Sigma)d\theta$$

PROBLEM

- **estimate parameters:** maximize over α and Σ the likelihood of data y_{obs}, marginalized over random effects β,

$$\pi(y_{\text{obs}} \mid \alpha, \Sigma) = \prod_i \int \pi(y_i \mid \theta)\pi(\theta \mid \alpha, \Sigma)d\theta$$

- **problem: evaluating these (high-dim) integrals is challenging**, especially with many individuals



AN AMORTIZED APPROACH

AN AMORTIZED APPROACH

- **idea:** rewrite in terms of an **individual-specific posterior:**

$$\pi(y_{\text{obs}} \mid \alpha, \Sigma) = \prod_i \pi(y_i) \int \pi(\theta \mid y_i) \frac{\pi(\theta \mid \alpha, \Sigma)}{\pi(\theta)} d\theta$$

$$= \prod_i \pi(y_i) \mathrm{E}_{\theta \sim \pi(\theta \mid y_i)} \left[\frac{\pi(\theta \mid \alpha, \Sigma)}{\pi(\theta)} \right]$$

AN AMORTIZED APPROACH

- **idea:** rewrite in terms of an **individual-specific posterior:**

$$\pi(y_{\text{obs}} \mid \alpha, \Sigma) = \prod_i \pi(y_i) \int \pi(\theta \mid y_i) \frac{\pi(\theta \mid \alpha, \Sigma)}{\pi(\theta)} d\theta$$

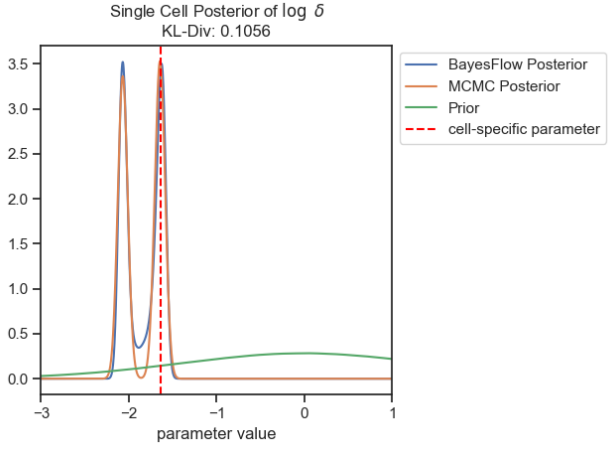
$$= \prod_i \pi(y_i) \mathrm{E}_{\theta \sim \pi(\theta \mid y_i)} \left[\frac{\pi(\theta \mid \alpha, \Sigma)}{\pi(\theta)} \right]$$

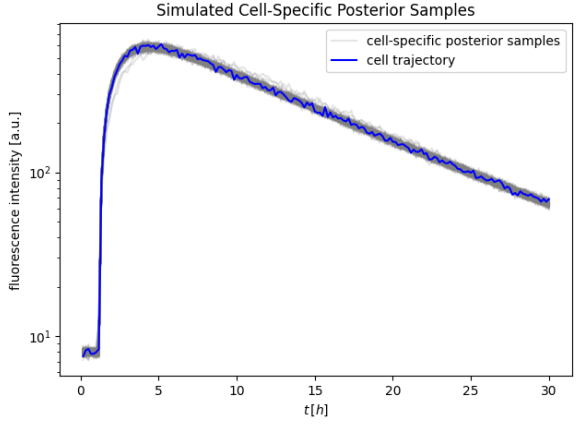
- ... and approximate the posterior using a **neural density estimator** trained on synthetic data!

EVALUATION

✓ INN **fits the posterior** well
✓ after training once (h), the optimization is **very fast** ($s - min$)
✓ allows to **easily test hypotheses**
✓ allows considering **stochastic models**
✓ even **uncertainty analysis** easily possible

EVALUATION





✓ INN **fits the posterior** well

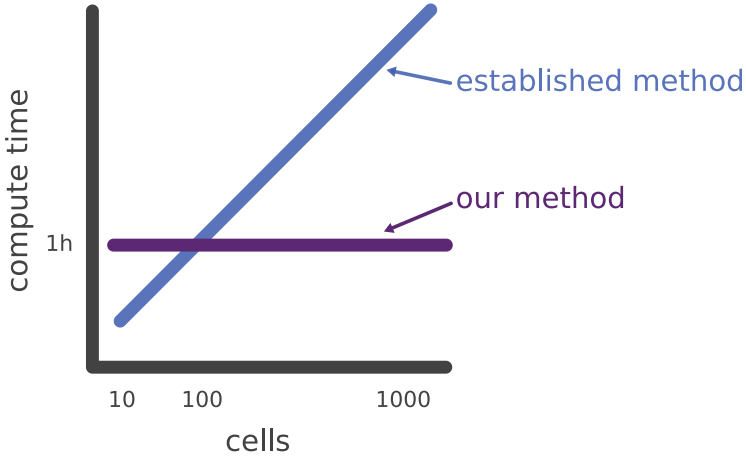
✓ after training once (h), the optimization is **very fast** ($s - min$)

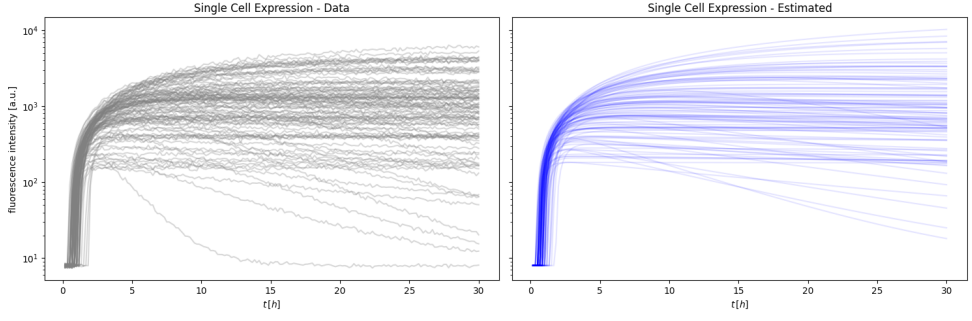
✓ allows to **easily test hypotheses**

✓ allows considering **stochastic models**

✓ even **uncertainty analysis** easily possible

EVALUATION





✓ INN **fits the posterior** well

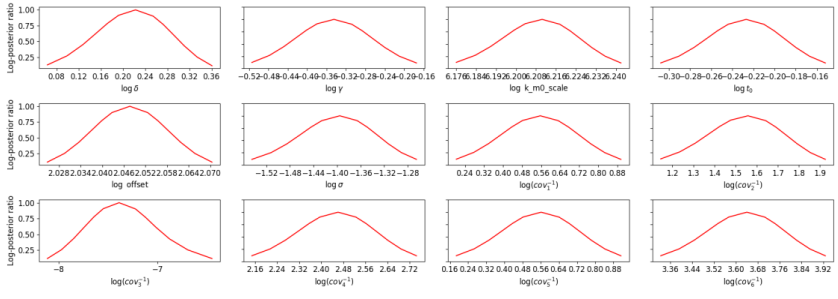
✓ after training once (h), the optimization is **very fast** ($s - min$)

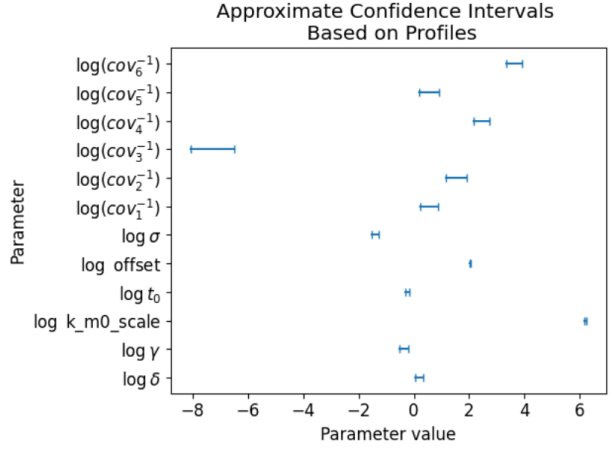
✓ allows to **easily test hypotheses**

✓ allows considering **stochastic models**

✓ even **uncertainty analysis** easily possible

EVALUATION





✓ INN **fits the posterior** well

✓ after training once (h), the optimization is **very fast** ($s - min$)

✓ allows to **easily test hypotheses**

✓ allows considering **stochastic models**

✓ even **uncertainty analysis** easily possible

OUTLOOK

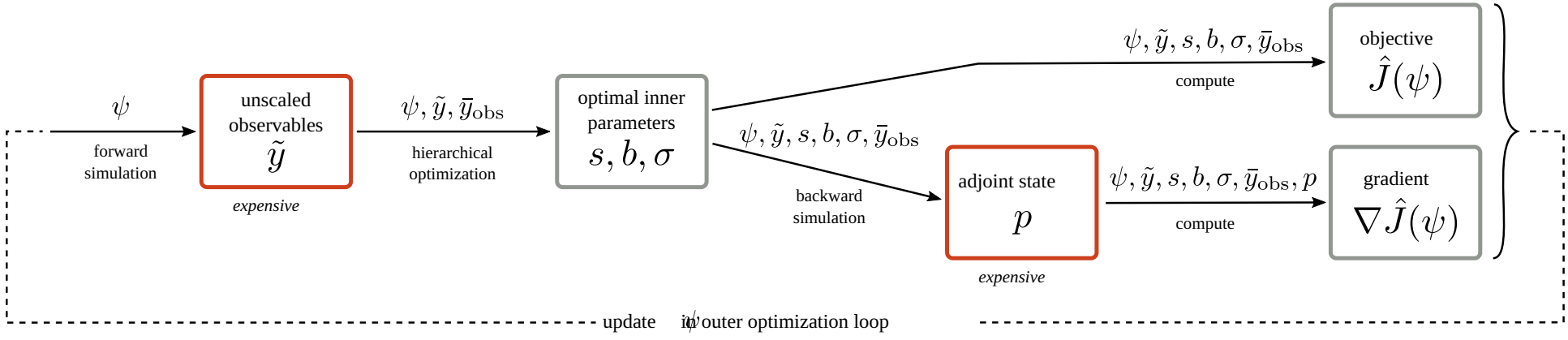
OUTLOOK

- applications in pharmacology and single-cell biology
- mechanistic insights to inform drug therapy
- combine with equation learning
- integrate image and tabular data in one framework
- large-scale modeling
- federated learning
- software


THANKS! QUESTIONS?
yannik-schaelte
@yannik_schaelte
yannikschaelte

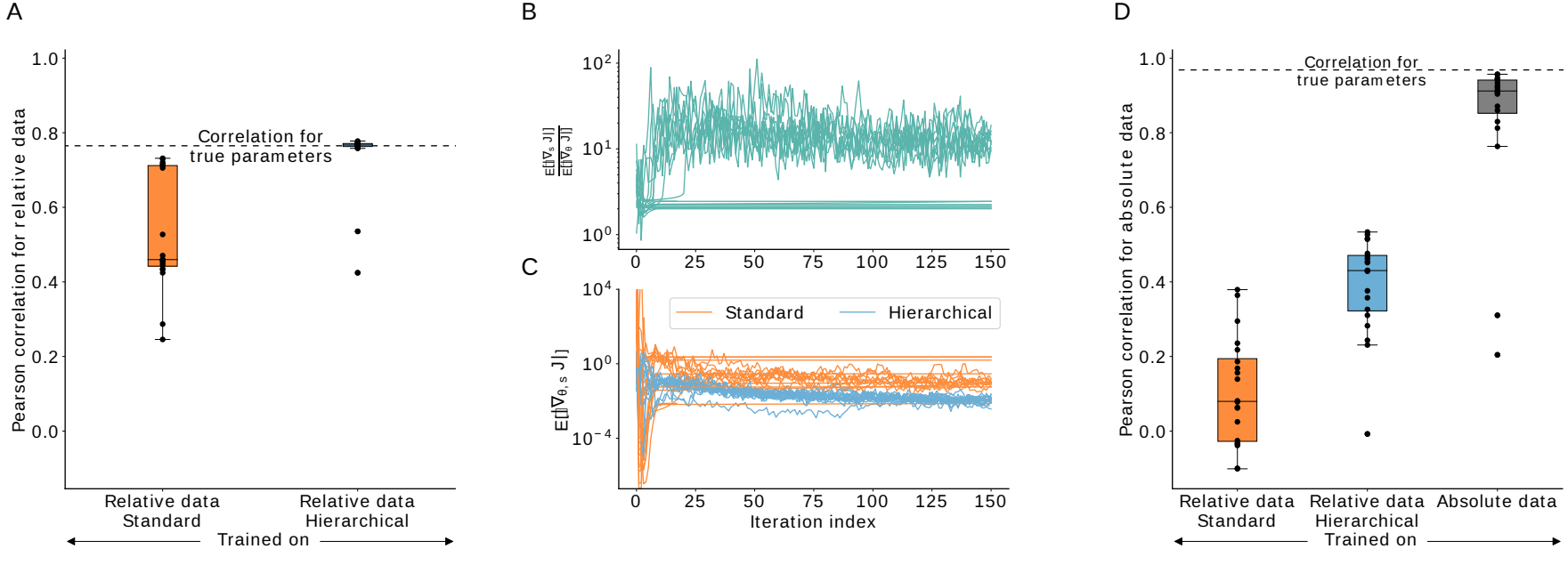
BACKUP

ILLUSTRATION OF ADJOINT-HIERARCHICAL EVALUATION SCHEME



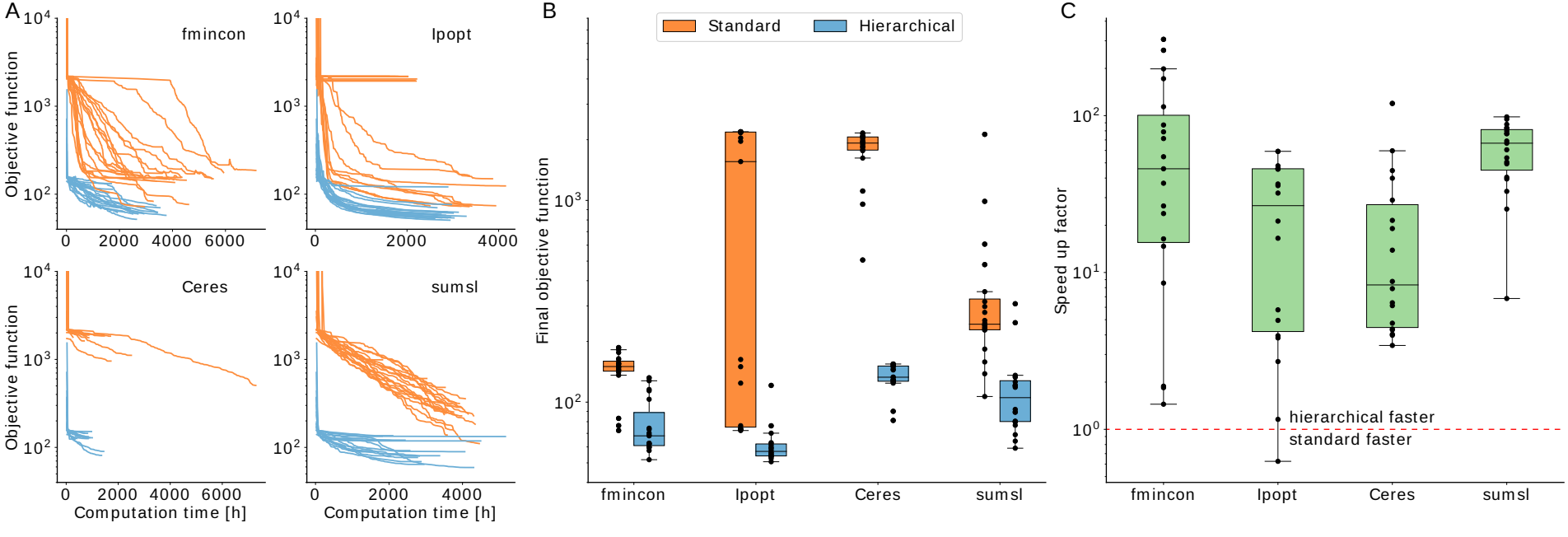
from: Schmiester, Schälte et al., Bioinformatics 2020

CONVERGENCE OF STANDARD AND HIERARCHICAL OPTIMIZATION



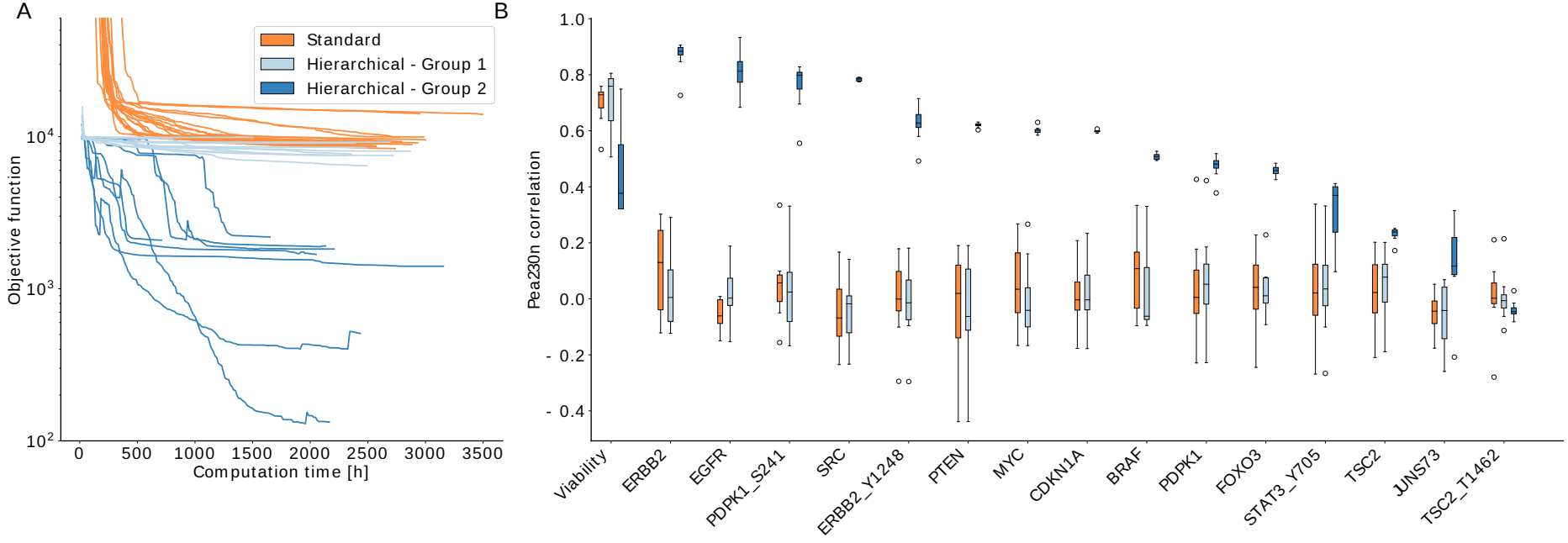
from: Schmiester, Schälte et al., Bioinformatics 2020

COMPUTATIONAL EFFICIENCY OF STANDARD AND HIERARCHICAL OPTIMIZATION



from: Schmiester, Schälte et al., Bioinformatics 2020

INTEGRATION OF HETEROGENEOUS DATA USING HIERARCHICAL OPTIMIZATION



from: Schmiester, Schälte et al., Bioinformatics 2020

pyABC

github.com/icb-dcm/pyabc

Klinger et al., Bioinformatics 2018 and Schälte et al., JOSS 2022

```python
# specify problem and parallelization
abc = ABCSMC(model, prior, distance, sampler)
# pass data
abc.new(db, data)
# run it
abc.run()
```

flexible

modular
configure
extend

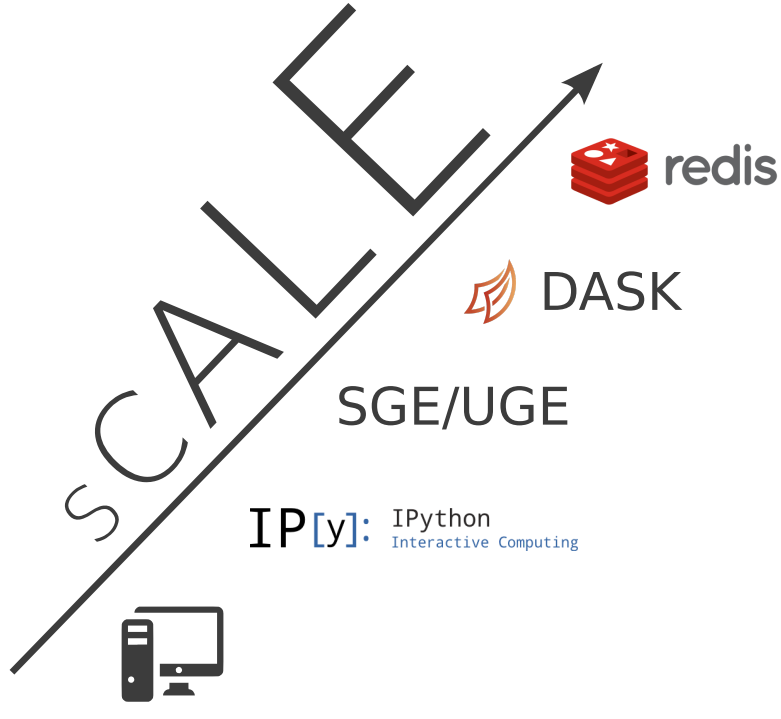
user-friendly

documented
self-tuning
robust

scalable

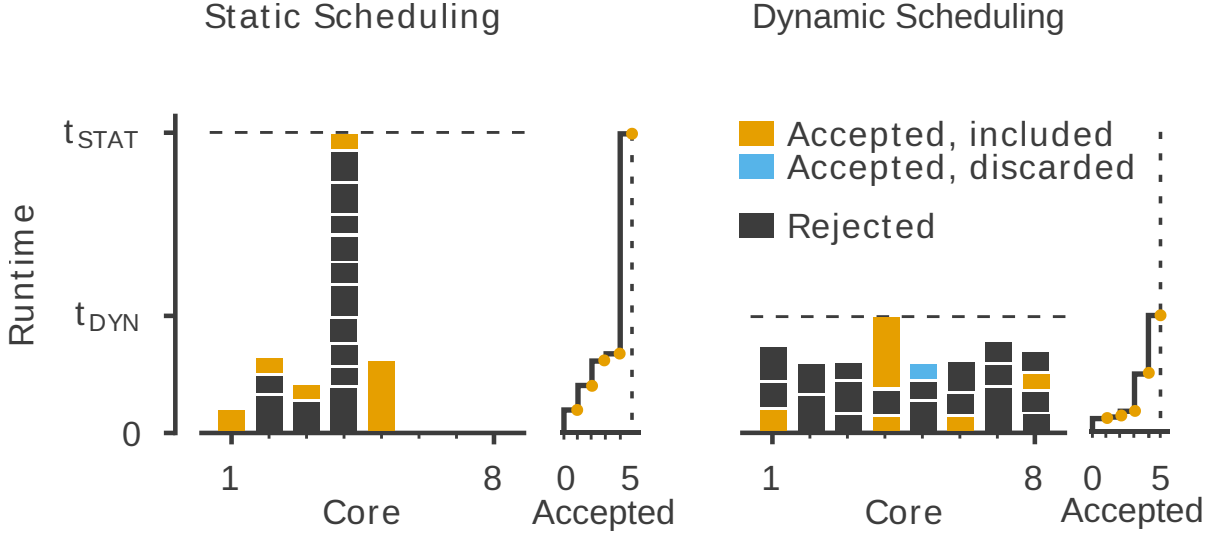
1 to 1,000s cores
efficient
tested

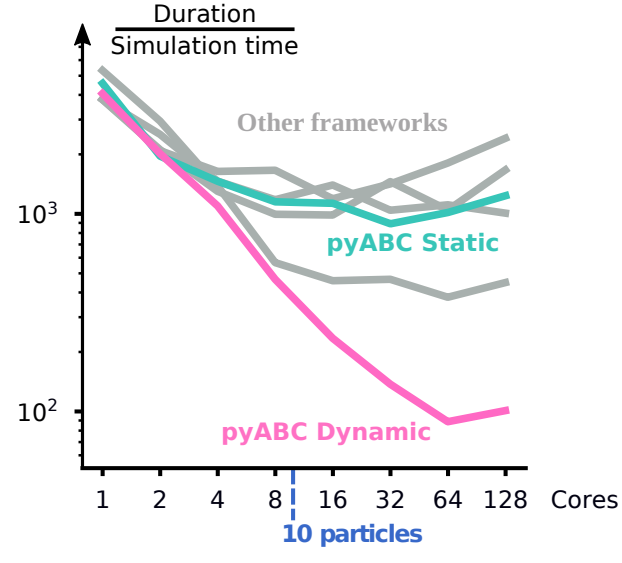
PARALLEL BACKENDS: 1 TO 1,000S CORES



PARALLELIZATION STRATEGIES

Klinger et al., CMSB Proceedings 2017

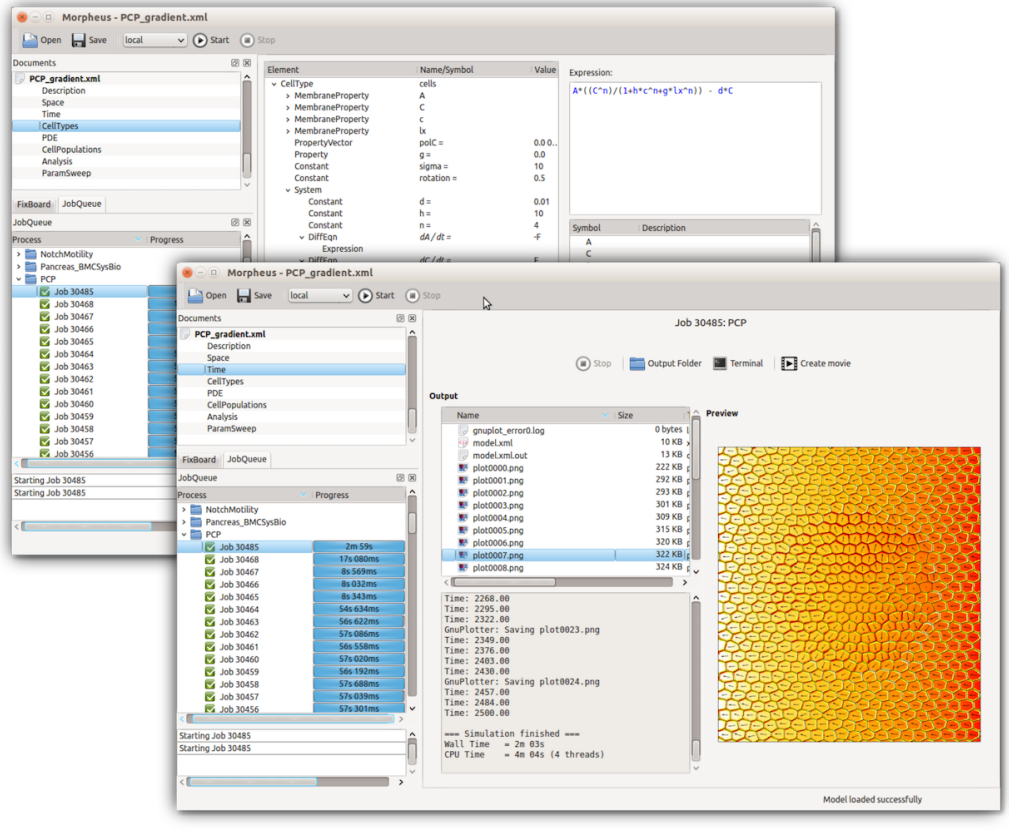




FitMultiCell

fitmulticell.gitlab.io

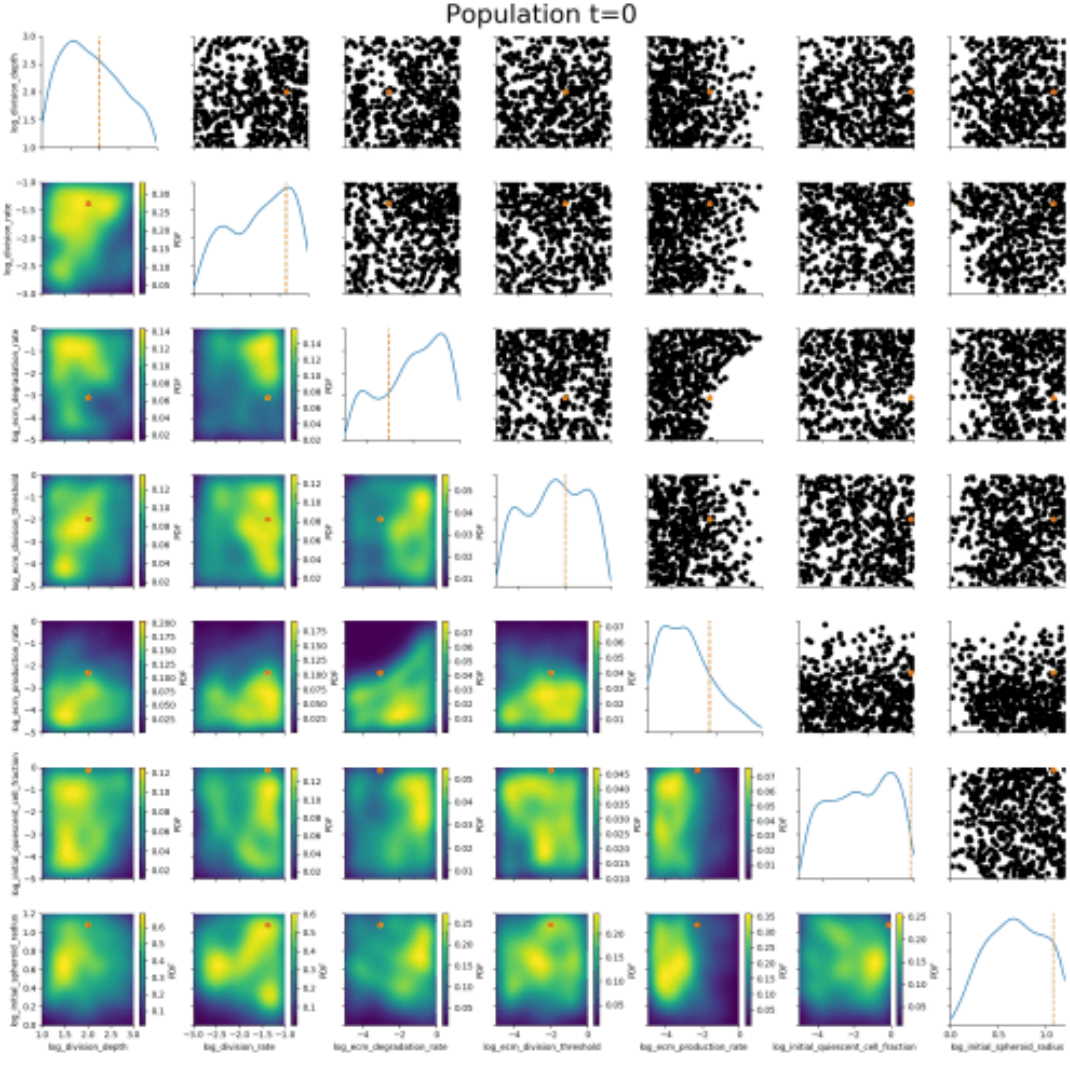
a platform for modeling, simulation and inference for multi-scale multi-cellular



models

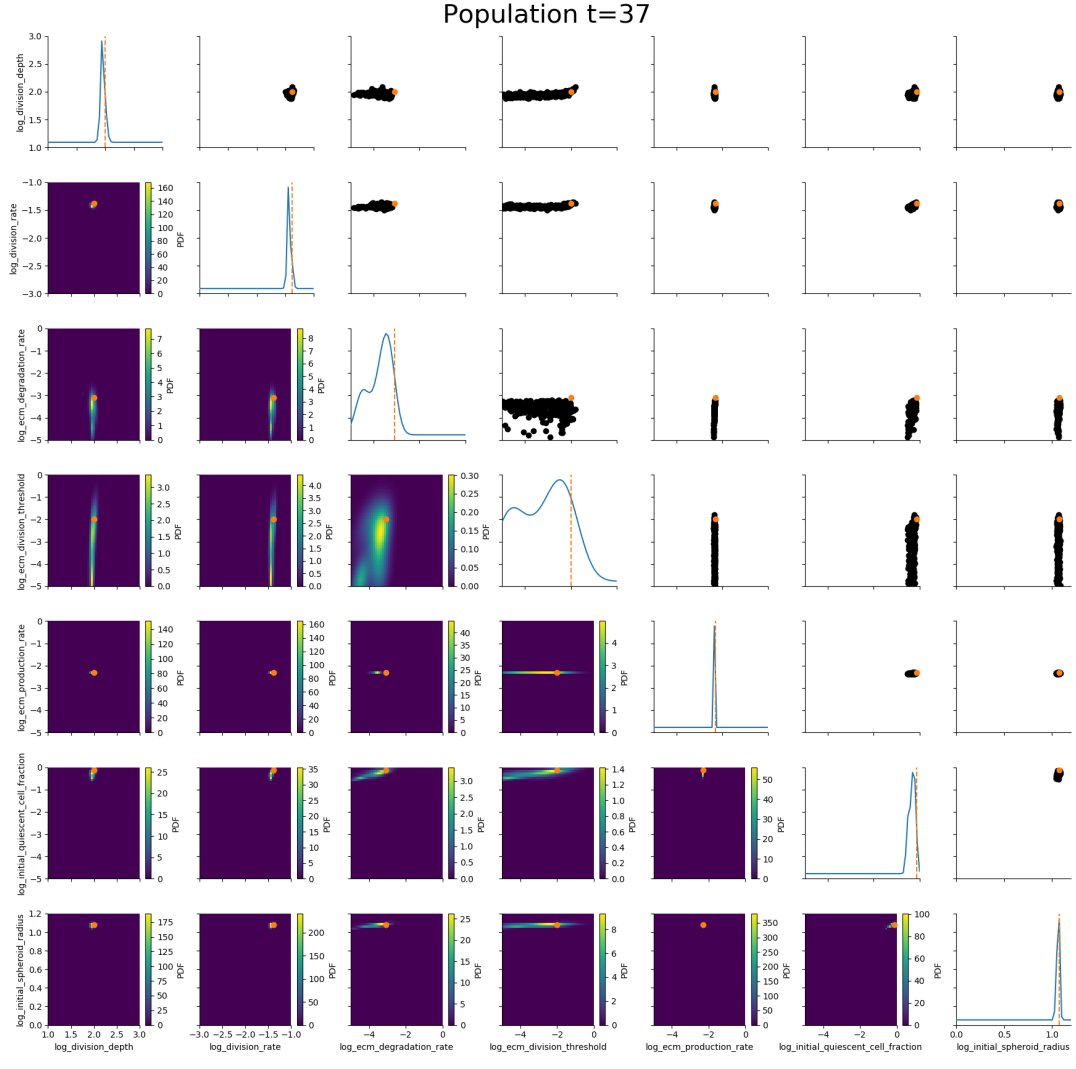
Starruß et al., Bioinformatics 2014; Alamoudi et al., NIC Proc. 2022; Alamoudi et al., bioRxiv 2023

APPLICATION EXAMPLE

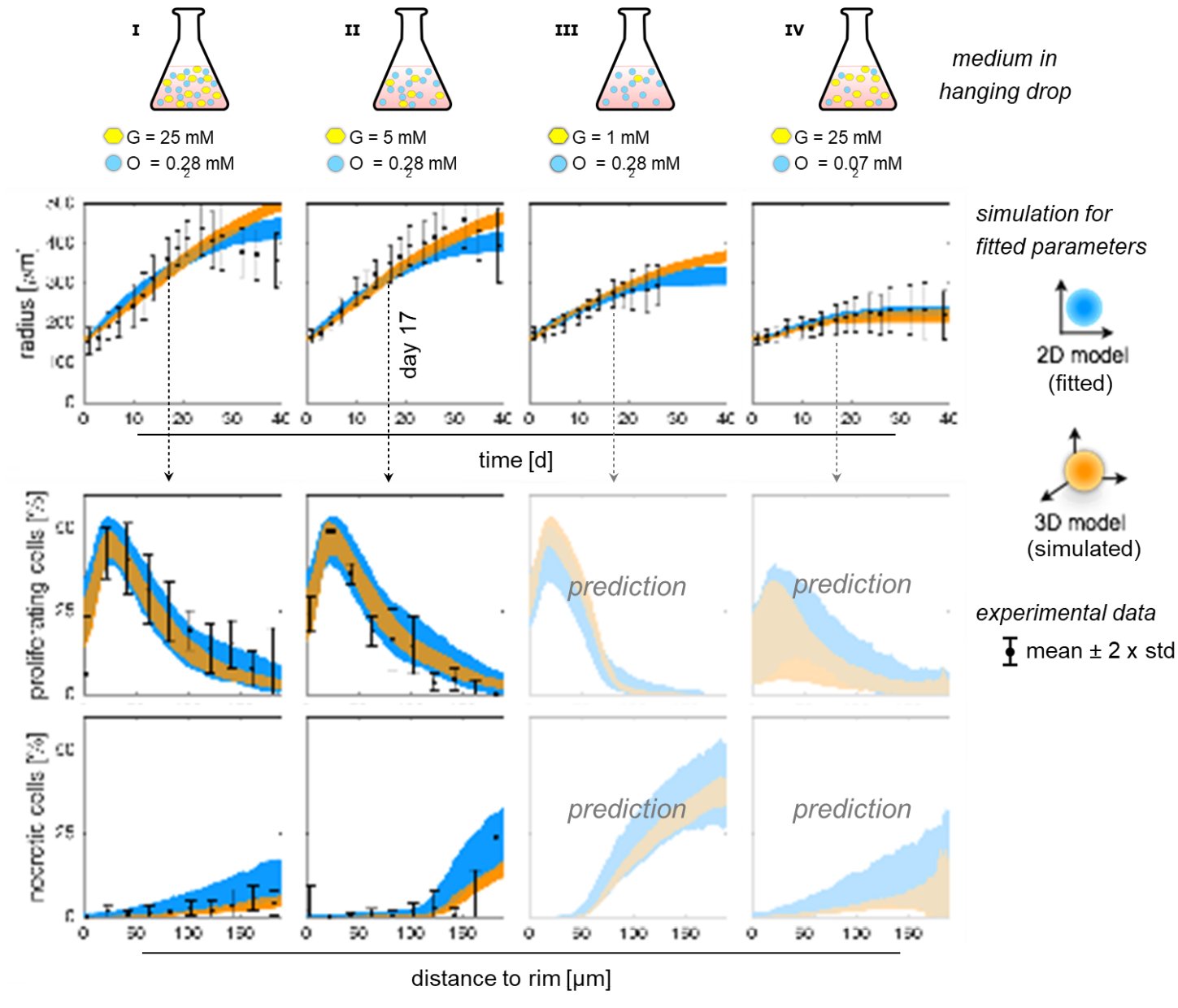


ABC worked where many other methods had failed.

APPLICATION EXAMPLE

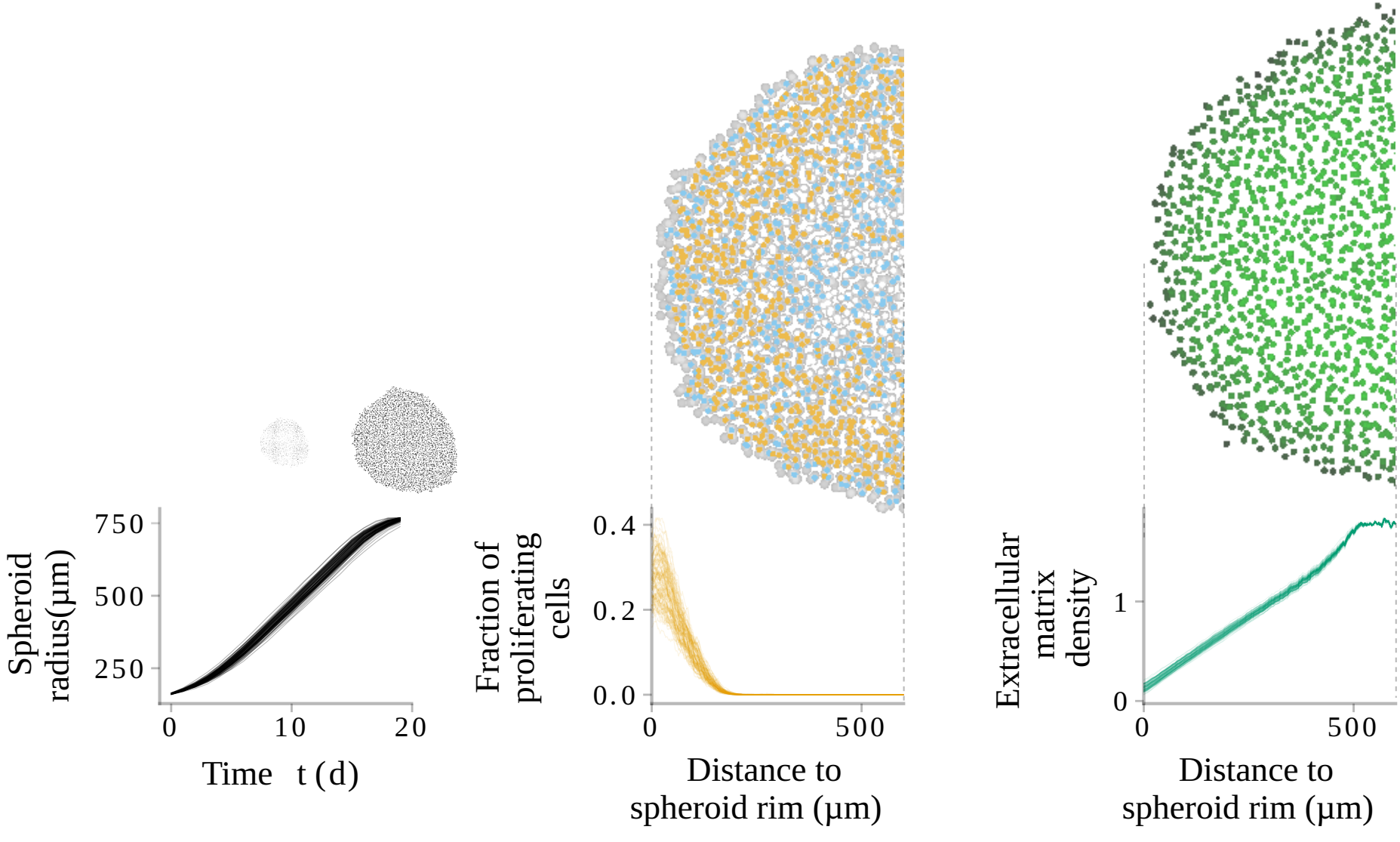


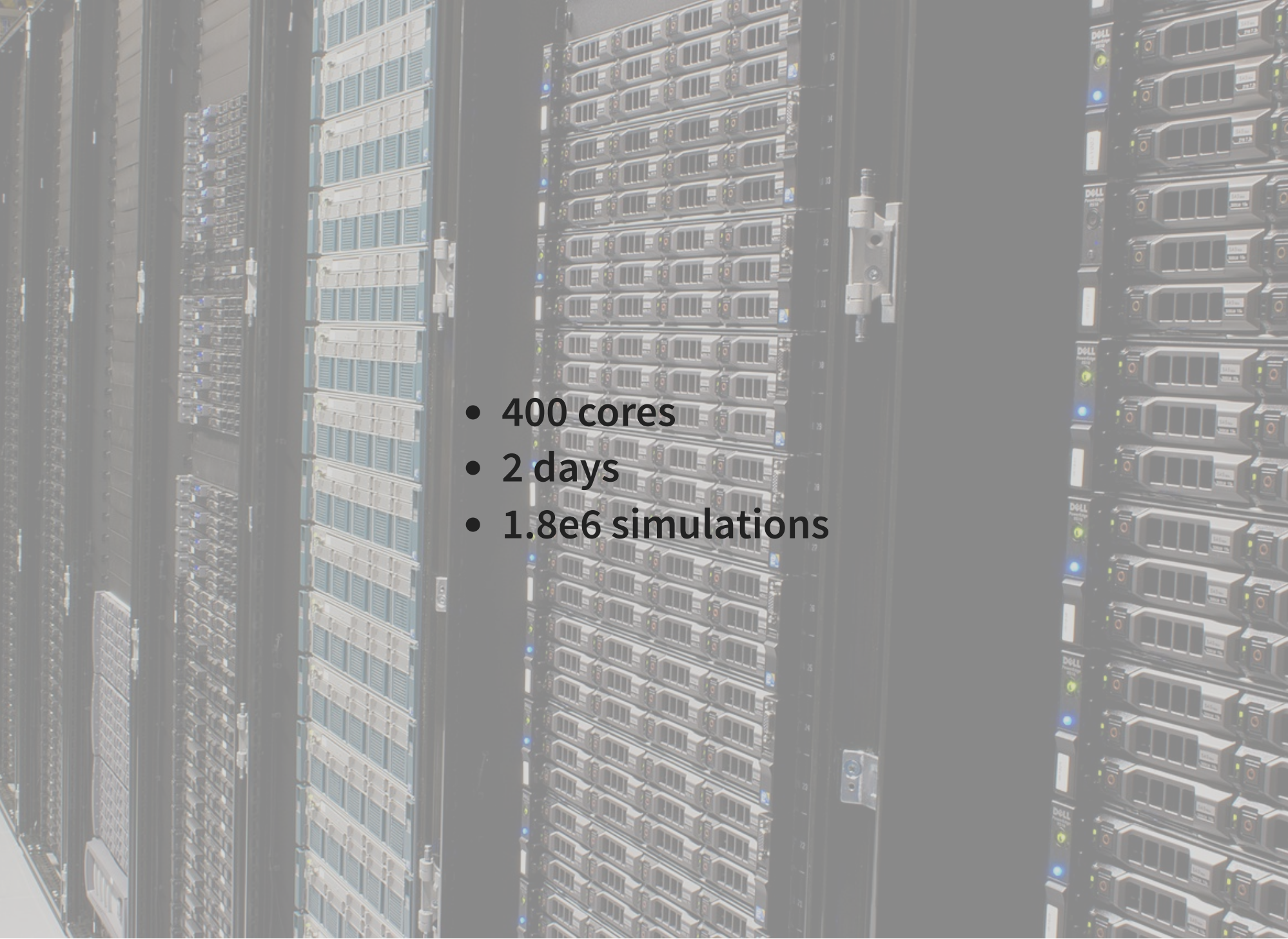
ABC worked where many other methods had failed.



Uncertainty-aware predictions, easy data integration.

DEFINE SUMMARY STATISTICS





- 400 cores
- 2 days
- 1.8e6 simulations

THEOREM (EXACT INFERENCE)

Using the **modified kernel** with $c > \pi(\bar{y}_{\text{obs}}|y, \theta)\ \forall y, \theta$, we sample from the **true posterior**

$$\pi_{\text{ABC}}(\theta|\bar{y}_{\text{obs}}) = \pi(\theta|\bar{y}_{\text{obs}}) \propto \int \pi(\bar{y}_{\text{obs}}|y, \theta)p(y|\theta)\, dy \cdot \pi(\theta)$$

assuming **noisy data** $\bar{y}_{\text{obs}} \sim \pi(\bar{y}|y, \theta)$.

- non-trivial **noise** allows to do **exact likelihood-free** inference
- applicable to **stochastic models**
- **parameterized noise** model

THEOREM (OPTIMAL SUMMARY STATISTICS)

[...] Given $\lambda : \mathbb{R}^{n_\theta} \to \mathbb{R}^{n_\lambda}$ such that $\mathbb{E}_{\pi(\theta)}[|\lambda(\theta)|] < \infty$, define **summary statistics as the conditional expectation**

$$s(y) := \mathbb{E}[\lambda(\Theta)|Y = y] = \int \lambda(\theta)\pi(\theta|y)d\theta.$$

Then, it holds $\left\|\mathbb{E}_{\pi_{\mathrm{ABC},\varepsilon}}[\lambda(\Theta)|s(y_{\mathrm{obs}})] - s(y_{\mathrm{obs}})\right\| \leq \varepsilon$, and therefore

$$\lim_{\varepsilon \to 0} \mathbb{E}_{\pi_{\mathrm{ABC},\varepsilon}}[\lambda(\Theta)|s(y_{\mathrm{obs}})] = \mathbb{E}[\lambda(\Theta)|Y = y_{\mathrm{obs}}].$$

In practice: Train regression model $s : y \mapsto \lambda(\theta) = (\theta^1, \ldots, \theta^k)$.

SCALE-NORMALIZING AND OUTLIER-ROBUST ADAPTIVE DISTANCES