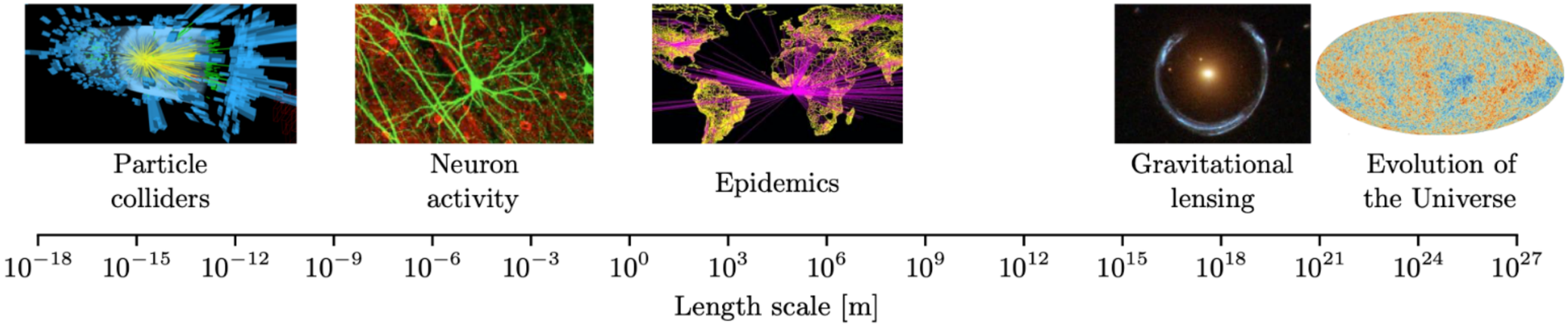
Benchmarking Simulation-Based Inference

Jan-Matthis Lueckmann
Machine Learning in Science
Prof. Dr. Jakob H. Macke

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Science uses simulators across all length scales



———

Cranmer et al., 2019; Images: CMS collaboration (2015); Lee WCA, et al. (2005); Research I (2019); NASA/ESA (2011)

Statistical inference for models with intractable likelihoods

COLLOQUIUM PAPER

The frontier of simulation-based inference

Kyle Cranmer[a,b,1], Johann Brehmer[a,b], and Gilles Louppe[c]

[a]Center for Cosmology and Particle Physics, New York University, New York, NY 10003; [b]Center for Data Science, New York University, New York, NY 10011; and [c]Montefiore Institute, University of Liège, B-4000 Liège, Belgium

Edited by Jitendra Malik, University of California, Berkeley, CA, and approved April 10, 2020 (received for review November 4, 2019)

Many domains of science have developed complex simulations to describe phenomena of interest. While these simulations provide high-fidelity models, they are poorly suited for inference and lead to challenging inverse problems. We review the rapidly developing field of simulation-based inference and identify the forces giving additional momentum to the field. Finally, we describe how the frontier is expanding so that a broad audience can appreciate the profound influence these developments may have on science.

statistical inference | implicit models | likelihood-free inference | approximate Bayesian computation | neural density estimation

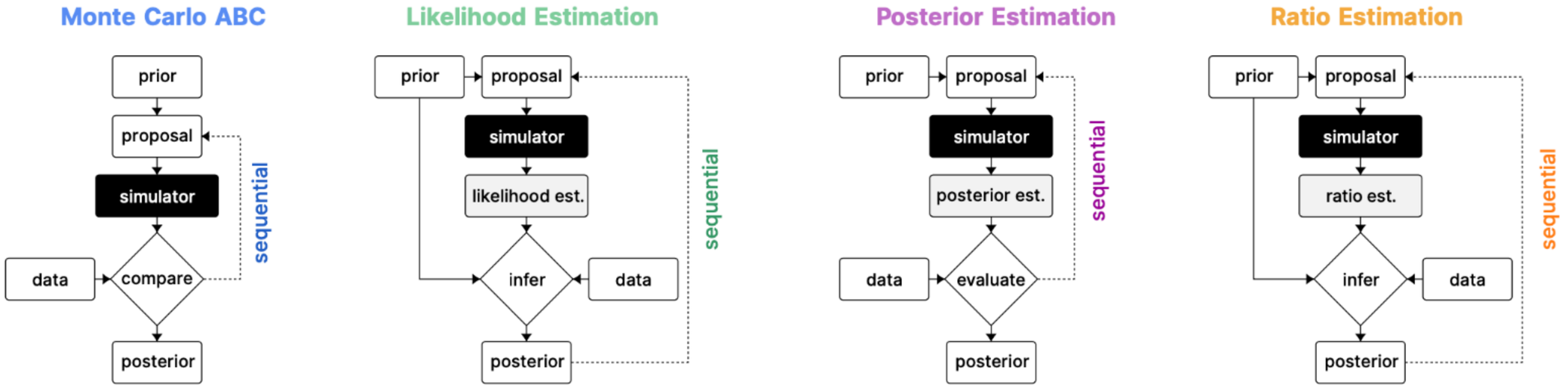
the simulator—is being recognized as a key idea to improve the sample efficiency of various inference methods. A third direction of research has stopped treating the simulator as a black box and focused on integrations that allow the inference engine to tap into the internal details of the simulator directly.

Amidst this ongoing revolution, the landscape of simulation-based inference is changing rapidly. In this review we aim to provide the reader with a high-level overview of the basic ideas behind both old and new inference techniques. Rather than discussing the algorithms in technical detail, we focus on the current frontiers of research and comment on some ongoing developments that we deem particularly exciting.

Rapid and exciting developments

However, papers often use different tasks and different metrics

Unified benchmark with algorithms, ...



Lueckmann, Boelts, Greenberg, Goncalves, & Macke (2021). Benchmarking simulation-based inference. Schemes after Cranmer et al. (2020).

Unified benchmark with algorithms, tasks, ...

T Tasks

T.1 Gaussian Linear

Inference of the mean of a 10-d Gaussian model, in which the covariance is fixed. The (conjugate) prior is Gaussian:

Prior $\mathcal{N}(\mathbf{0}, 0.1 \odot \mathbf{I})$

Simulator $\mathbf{x}|\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{x}|\mathbf{m}_{\boldsymbol{\theta}} = \boldsymbol{\theta}, \mathbf{S} = 0.1 \odot \mathbf{I})$

Dimensionality $\boldsymbol{\theta} \in \mathbb{R}^{10}, \mathbf{x} \in \mathbb{R}^{10}$

T.2 Gaussian Linear Uniform

Inference of the mean of a 10-d Gaussian model, in which the covariance is fixed. The prior is uniform:

Prior $\mathcal{U}(-\mathbf{1}, \mathbf{1})$

Simulator $\mathbf{x}|\boldsymbol{\theta} \sim \mathcal{N}(\mathbf{x}|\mathbf{m}_{\boldsymbol{\theta}} = \boldsymbol{\theta}, \mathbf{S} = 0.1 \odot \mathbf{I})$

Dimensionality $\boldsymbol{\theta} \in \mathbb{R}^{10}, \mathbf{x} \in \mathbb{R}^{10}$

T.3 SLCP

A challenging inference task designed to have a simple likelihood and a complex posterior. The prior is uniform over five parameters $\boldsymbol{\theta}$ and the data are a set of four two-dimensional points sampled from a Gaussian likelihood whose mean and variance are nonlinear functions of $\boldsymbol{\theta}$:

Prior $\mathcal{U}(-\mathbf{3}, \mathbf{3})$

Simulator $\mathbf{x}|\boldsymbol{\theta} = (\mathbf{x}_1, \ldots, \mathbf{x}_4)$, $\mathbf{x}_i \sim \mathcal{N}(\mathbf{m}_{\boldsymbol{\theta}}, \mathbf{S}_{\boldsymbol{\theta}})$,
where $\mathbf{m}_{\boldsymbol{\theta}} = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}$, $\mathbf{S}_{\boldsymbol{\theta}} = \begin{bmatrix} s_1^2 & \rho s_1 s_2 \\ \rho s_1 s_2 & s_2^2 \end{bmatrix}$, $s_1 = \theta_3^2, s_2 = \theta_4^2, \rho = \tanh \theta_5$

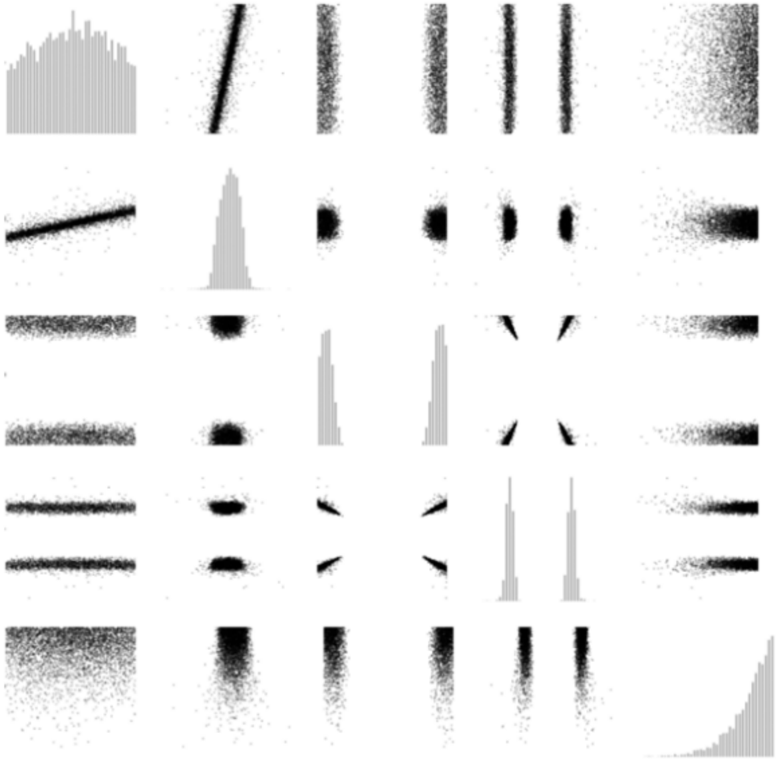
Dimensionality $\boldsymbol{\theta} \in \mathbb{R}^5, \mathbf{x} \in \mathbb{R}^8$

References Papamakarios et al. (2019b); Greenberg et al. (2019); Hermans et al. (2020)
Durkan et al. (2020)

T.4 SLCP with Distractors

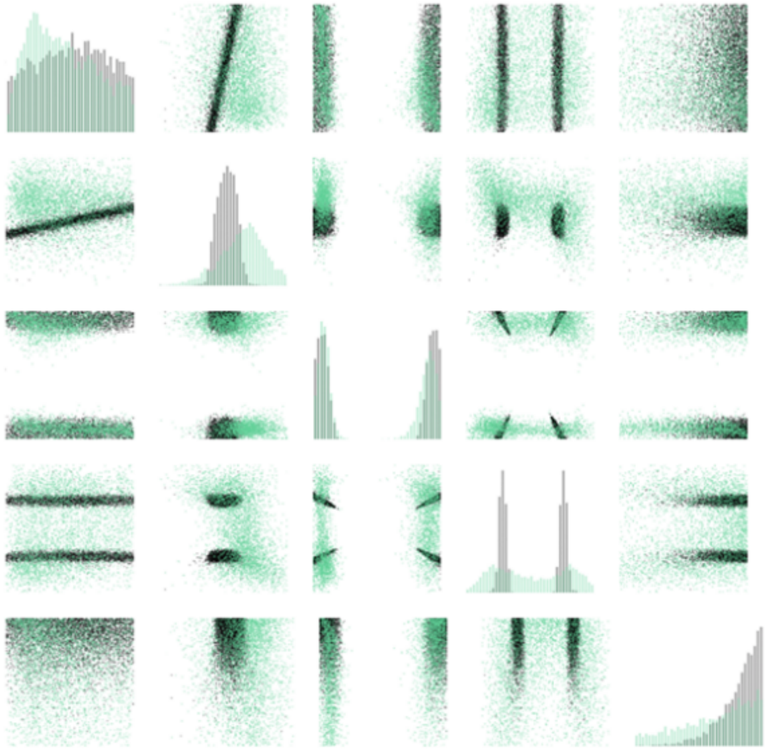
This task is similar to T.3, with the difference that we add uninformative dimensions (distractors) to the observation:

Prior $\mathcal{U}(-\mathbf{3}, \mathbf{3})$



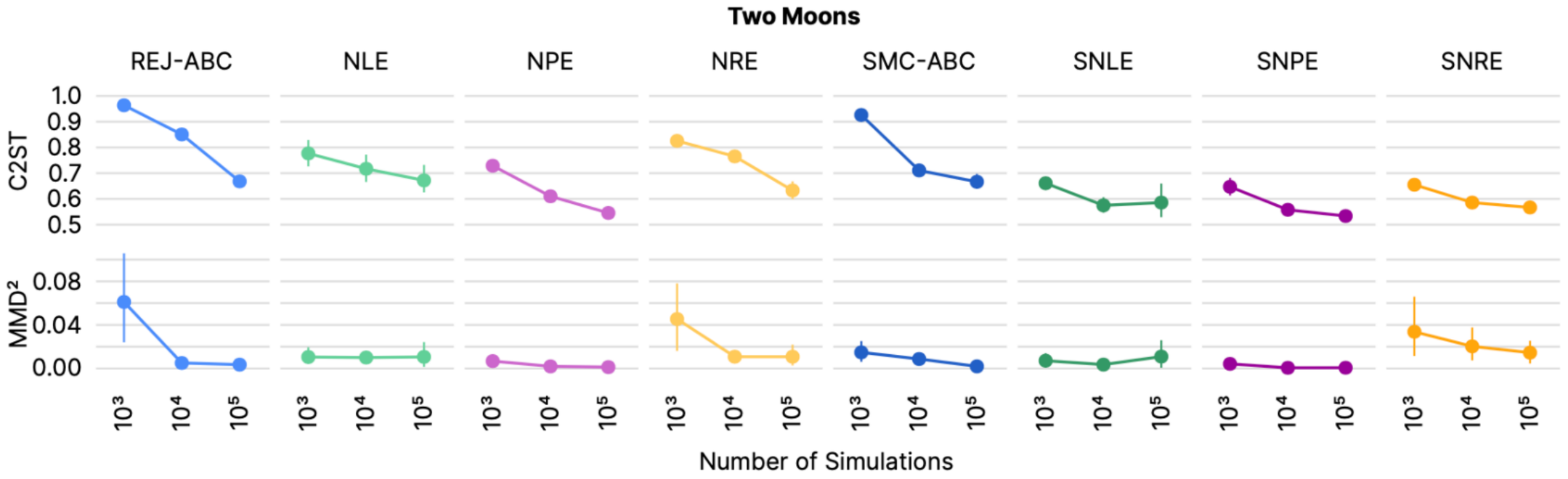
Unified benchmark with algorithms, tasks, ...

... and metrics

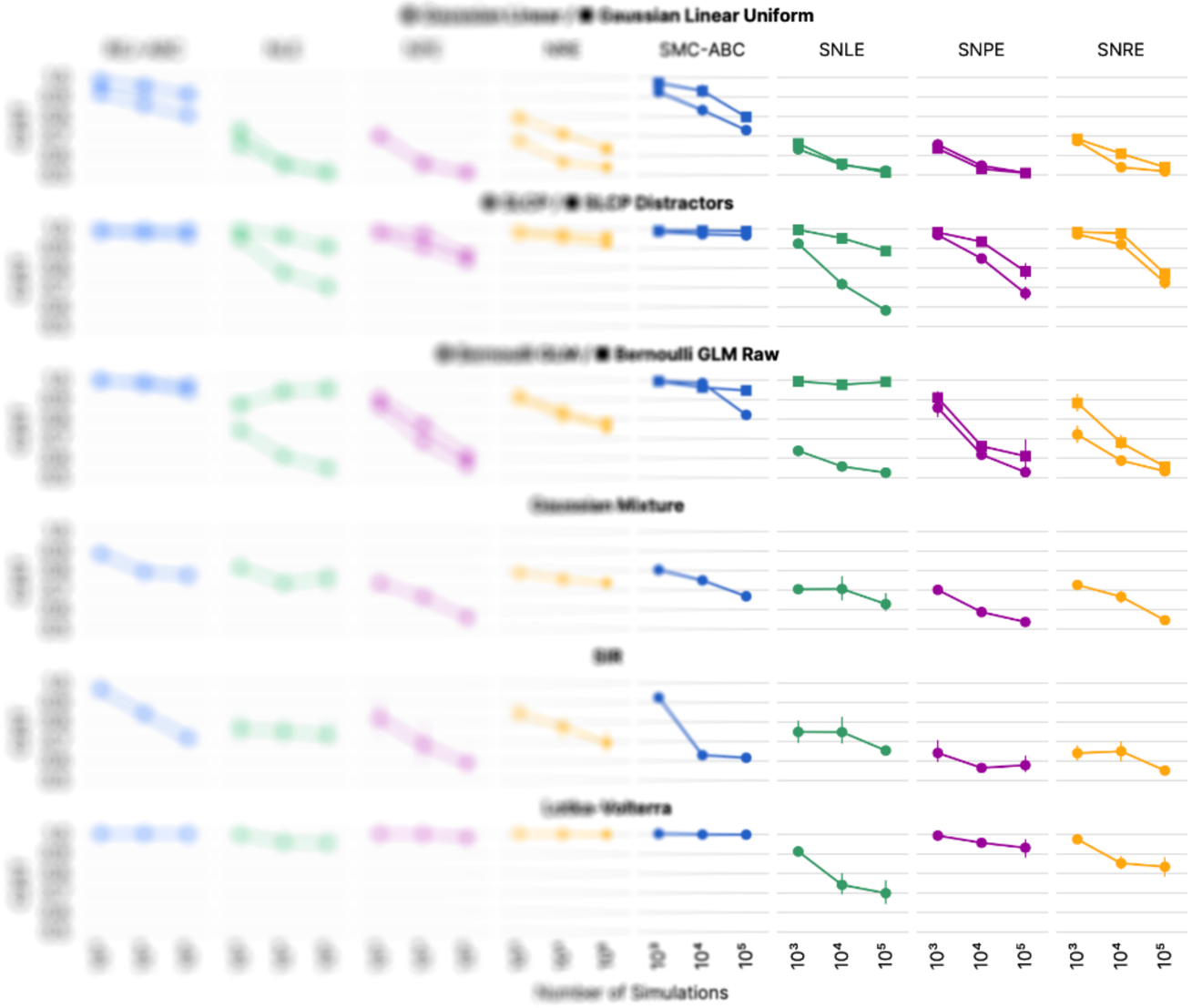


What did we learn from 10 000+ runs?

Performance metric is key



No uniformly best algorithm



Benchmark as a useful tool —
diagnose implementation issues
and improve algorithms

sbi-benchmark.github.io

Overview | Interactive Results | Code & Reproducibility | Contributions | Code | Paper

Dataset

Main paper

Task

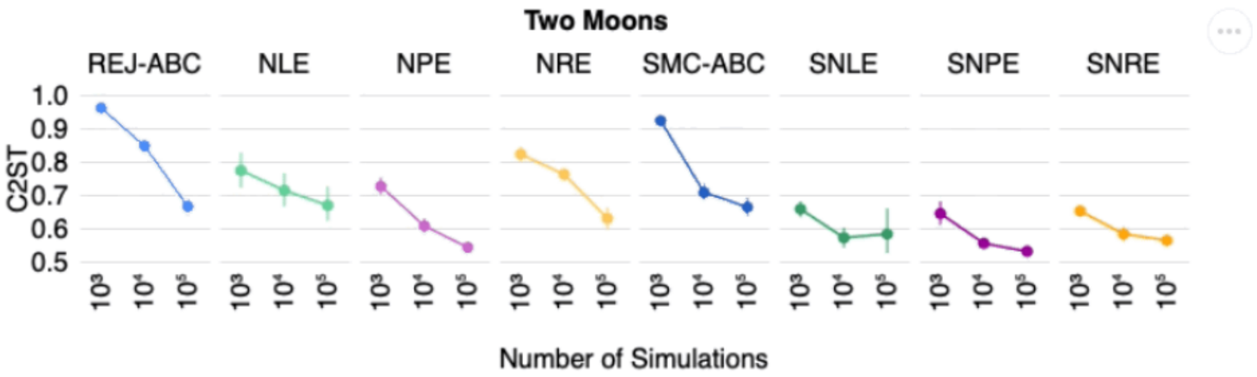
Two Moons

Metric

C2ST

C2ST
KSD
MEDDIST
MMD
NLTP
RT

Observations

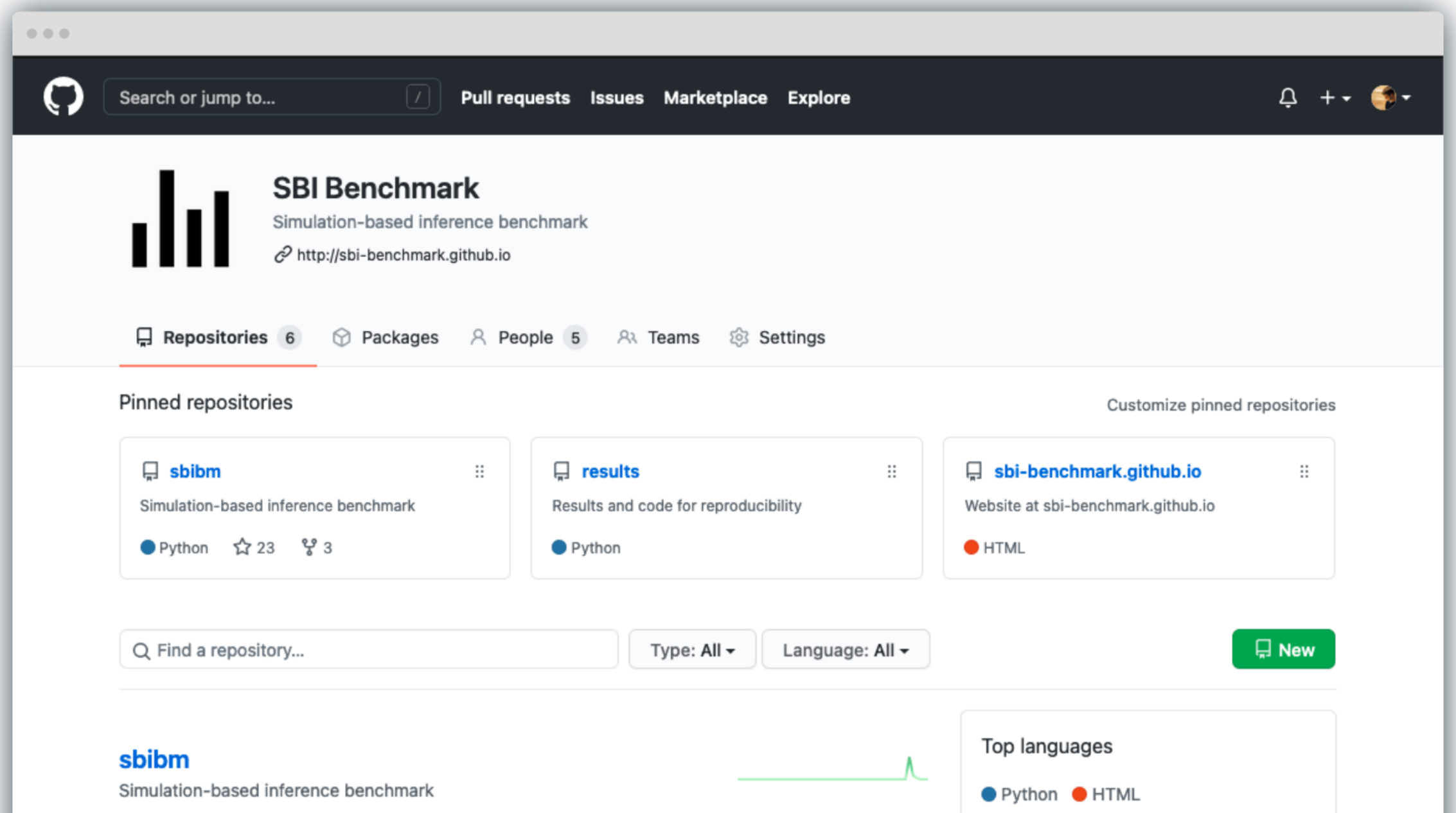


Comparing results of 240 runs in terms of C2ST (Classifier 2-sample test accuracy) on the Two Moons task. Error bars show 95% confidence intervals around the mean. Using the sidebar on the left you can change the selection.

Limitations

First step; we hope to inspire the community to participate!

All code and results available



All code and results available

README.md

Installation

Assuming you have a working Python environment, simply install `sbibm` via `pip`:

```
$ pip install sbibm
```

ODE based models (currently SIR and Lotka-Volterra models) use Julia via `diffeqtorch`. If you are planning to use these tasks, please additionally follow the installation instructions of `diffeqtorch`. If you are not planning to simulate these tasks for now, you can skip this step.

Quickstart

A quick demonstration of `sbibm`, see further below for more in-depth explanations:

```
import sbibm

task = sbibm.get_task("two_moons")  # See sbibm.get_available_tasks() for all tasks
prior = task.get_prior()
simulator = task.get_simulator()
observation = task.get_observation(num_observation=1)  # 10 per task

# These objects can then be used for custom inference algorithms, e.g.
# we might want to generate simulations by sampling from prior:
thetas = prior(num_samples=10_000)
xs = simulator(thetas)
```

Q&A

Practioners' advice

- Do we need the Bayesian posterior, or is a point estimate sufficient?
- Is the simulator really ‘black-box’?
- What domain knowledge do we have about the problem?
- Do we have, or can we learn summary statistics?
- Do we have low-dimensional data and parameters, and a cheap simulator?
- Are simulations expensive? Can we simulate online?
- Do we want to carry out inference once, or repeatedly?

See Lueckmann et al. (2021)