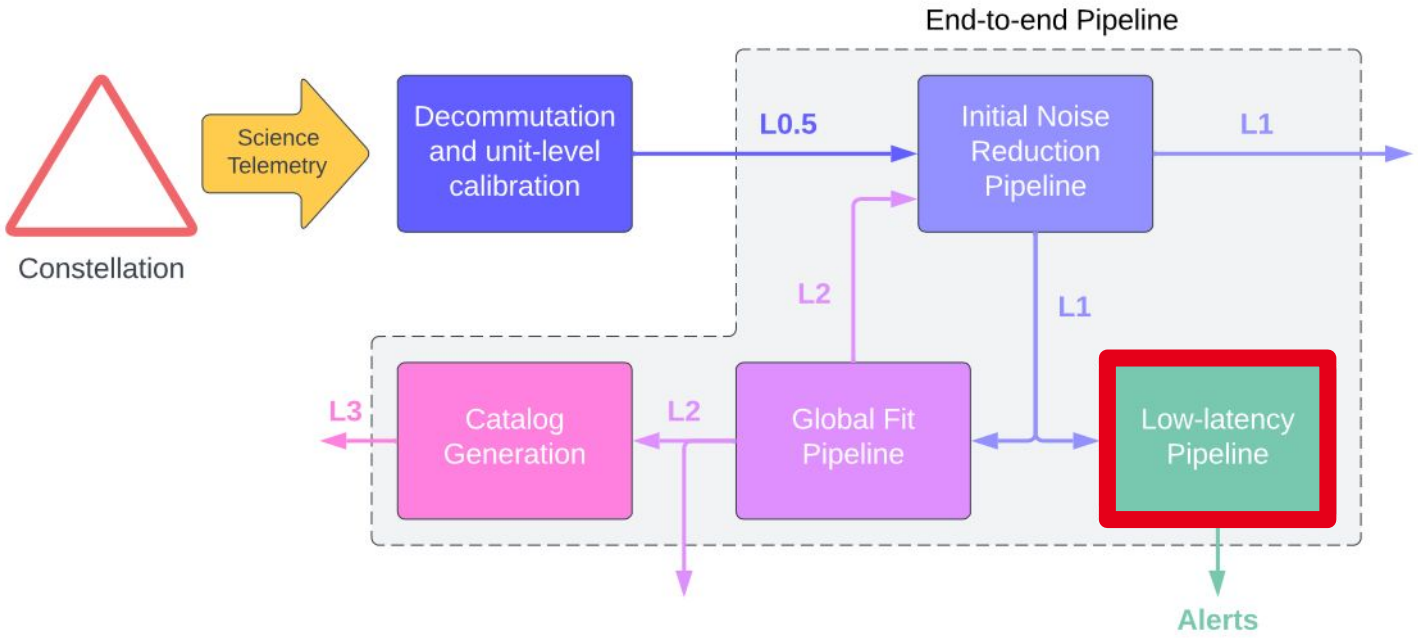
Simulation based inference (SBI) for fast parameter estimation of massive black hole binaries

Louis Le Saulnier, Tobías Liaudat, Jean-Baptiste Bayle

LISA data analysis and low latency alert pipeline

Objectif



The low latency alert pipeline:

Goal : send live alerts of MBHB events

Alert : tells within 1h after data reception if a MBHB has been detected and gives its parameter estimation (sky location)

Contribution to science objectives : helps the community to observe electromagnetic counterparts, supports multimessenger astronomy

Strong time constraints on our algorithms : **the method should take 20 minutes to perform parameter estimation**

Parameter estimation methods

Definition

- We aim estimate the parameters θ of massive black hole binaries (MBHB), especially the coalescence time and the sky location
- Here we call $\mathbf{x}$ the data, i.e. LISA data observations after pre-treatment (TDI data)

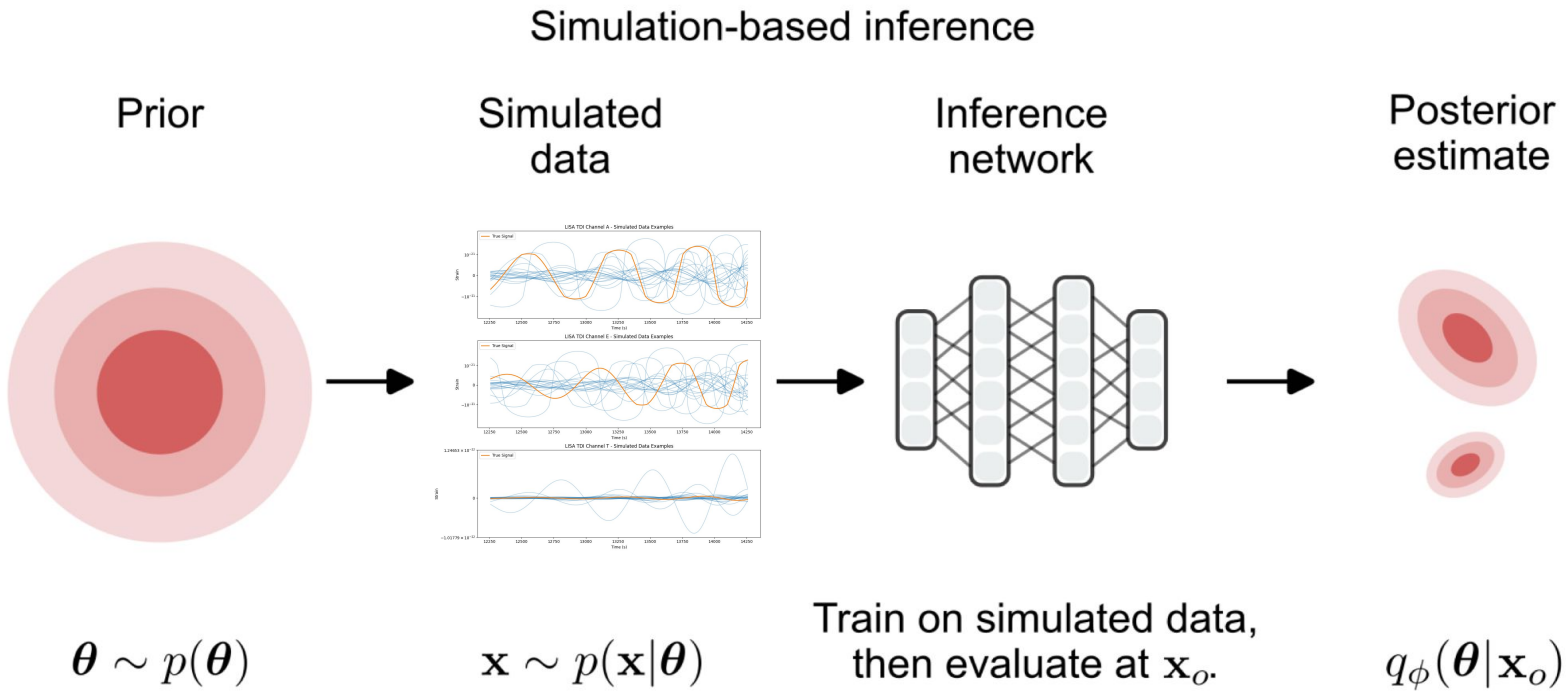
Bayesian Inference

- The goal is to find the posterior distribution of the parameters: $p(\theta|\mathbf{x}) \propto p(\mathbf{x}|\theta)p(\theta)$
  - $p(\theta)$ the prior distribution, mostly uniform laws on the parameters in our case
  - $p(\mathbf{x}|\theta)$ the likelihood function
- Based on Monte Carlo Markov Chain (MCMC) methods
- The posterior distribution gives parameter estimation and error bars
- Has convergence guarantees but **needs to evaluate a consequent amount of likelihoods**

Parameter estimation methods

Simulation based inference

- Training a Neural Posterior Estimator $q_\phi(\theta|\mathbf{x})$ that approximate the posterior distribution over the parameters
- Once the network is trained, the posterior is fully amortized, for any observation $\mathbf{x}_o$ the posterior $q_\phi(\theta|\mathbf{x}_o)$ can be obtained after one forward pass.



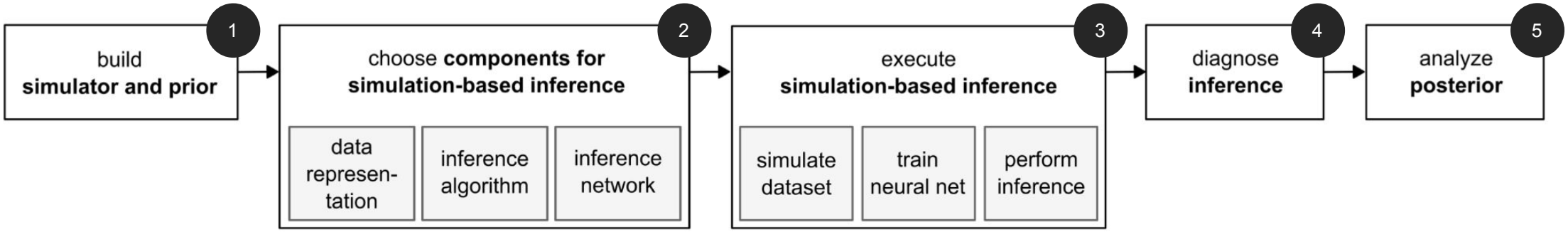
SBI workflow

Current goal

Build a full pipeline that performs posterior estimation using SBI but also deals with the needs of LISA data analysis and especially LLAP.

The following parts will be presented

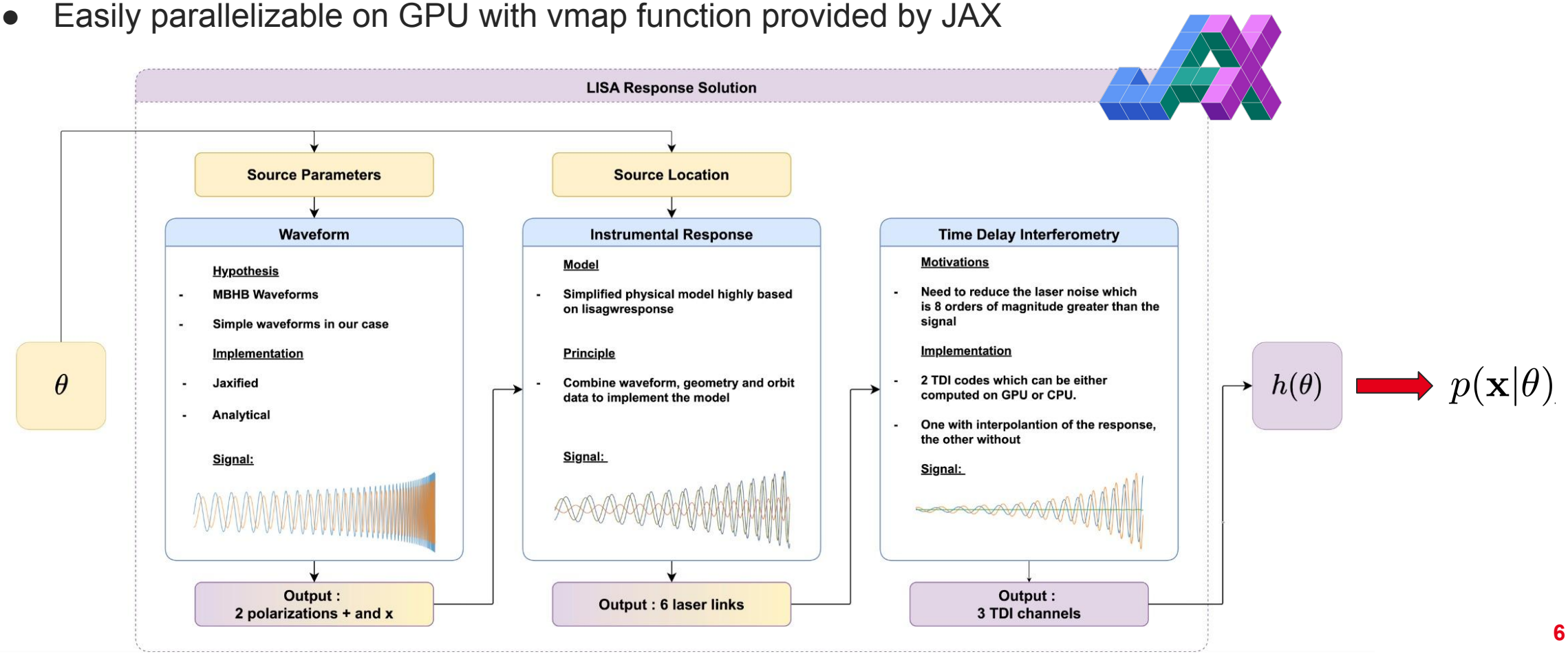
- The simulation part with a JAX LISA response and a dataset generator
- Data representation, how we choose summary statistics with LISA data ?
- GPU compatible training script and dealing with large datasets
- Results and diagnostics



LISA JAX response in time domain

Features:

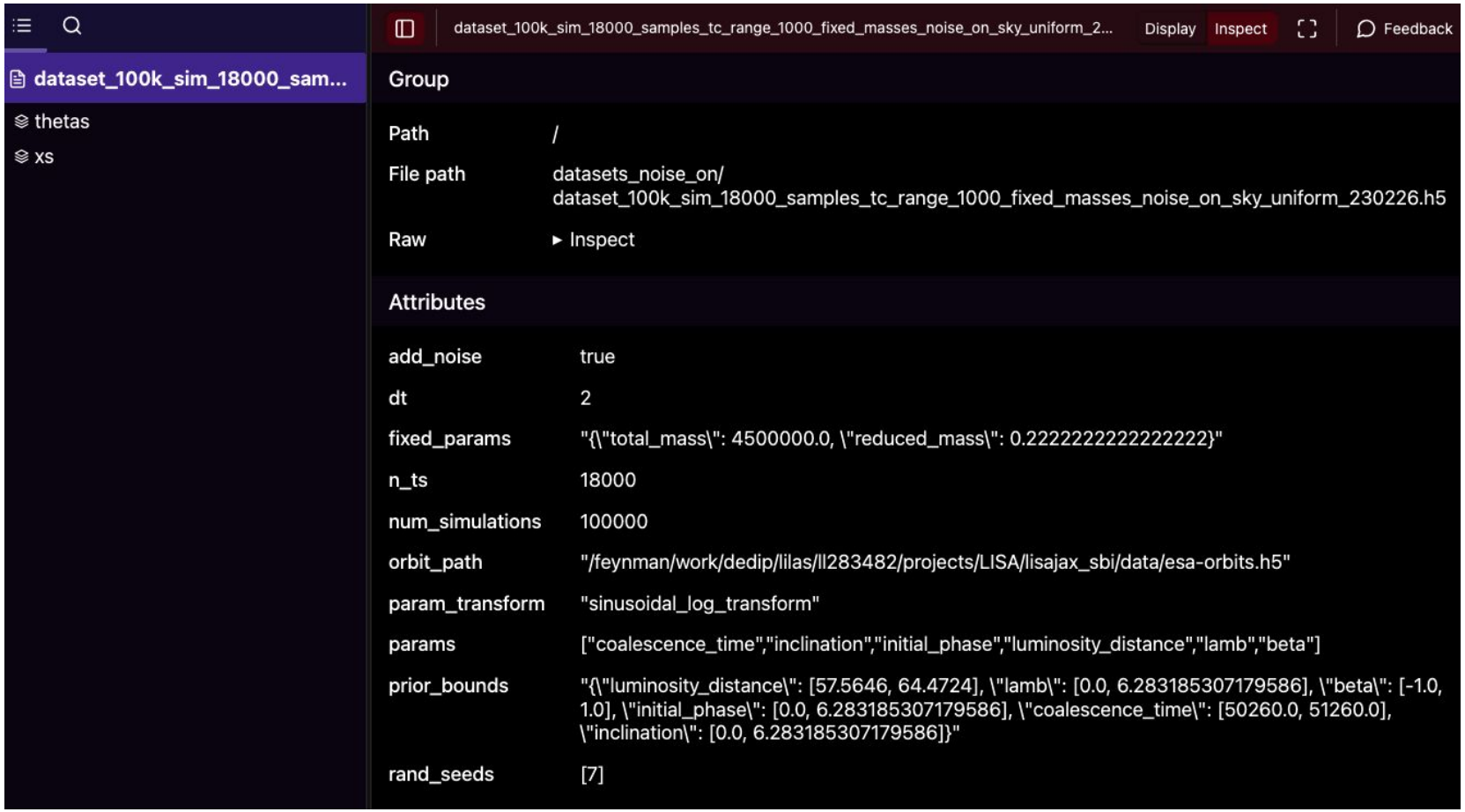
- Fully differentiable from waveform to likelihood
- Accelerated with JIT
- Easily parallelizable on GPU with vmap function provided by JAX



Dataset generation

From simulator to datasets:

- In SBI the simulator can be arbitrary, we only need to generate paired training datasets $\mathscr{D} = \{(\mathbf{x}_i, \theta_i)\}_{i=1,\dots,N}$
- Scripts to generate, extend and concatenate datasets if compatibles
- Stored in hdf5 files with all the attributes used to generate
- Fully reproducible data generation with seed management and attribute reading functions



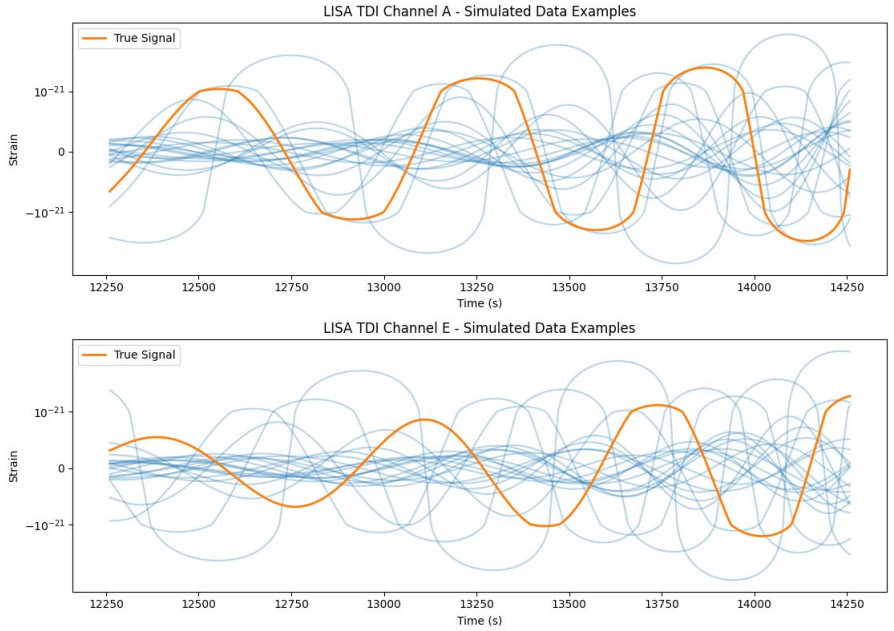
Summary statistics

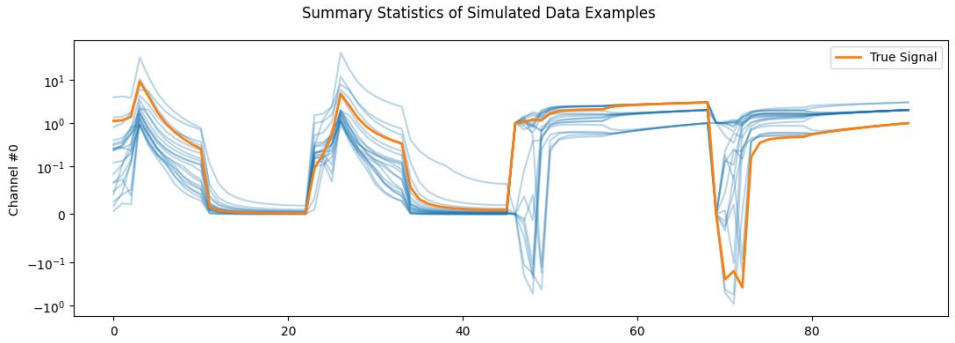
Motivation:

Neural networks struggle with high dimensional input data. As we are working with time series with more than 1000 datapoints there is a **need to find new data representations of our signals.** It can be:

- Analytical summary statistics (Fourier transform, wavelet transform,…)
- Embedding neural networks (CNN, RNN, transformer,...)

First idea: Concatenating the module and the phase of the Fourier transform of channels A and E





Training of the neural network

Training:

Once the simulator and the data representations are chosen, the SBI toolbox provides built in models and tools for training making it easy to compute

- The neural networks used are normalizing flows, which learn how to transform a simple reference distribution into a complex one (posterior here)
  - Density estimator in our work: **Masked Autoregressive Flow** (standard one in SBI)
  - Loss: $\mathscr{L}(\phi) = \frac{1}{N} \sum - \log q_\phi(\theta_i | \mathbf{x}_i) + C$
- The training can be performed either on GPU and CPU even with a large amount of data.
- SBI allows to train embedding networks during the training of the density estimator

Loss curves provided by SBI toolbox

Sampling of the posterior

Sampling:

When the model is trained the posterior can be built and then sampled for any data observation $\mathbf{x}_o$.

- Fast sampling of the posterior for any observation, meaning **fast parameter estimation.**
- However, the sampling can still be slightly improved

Importance sampling:

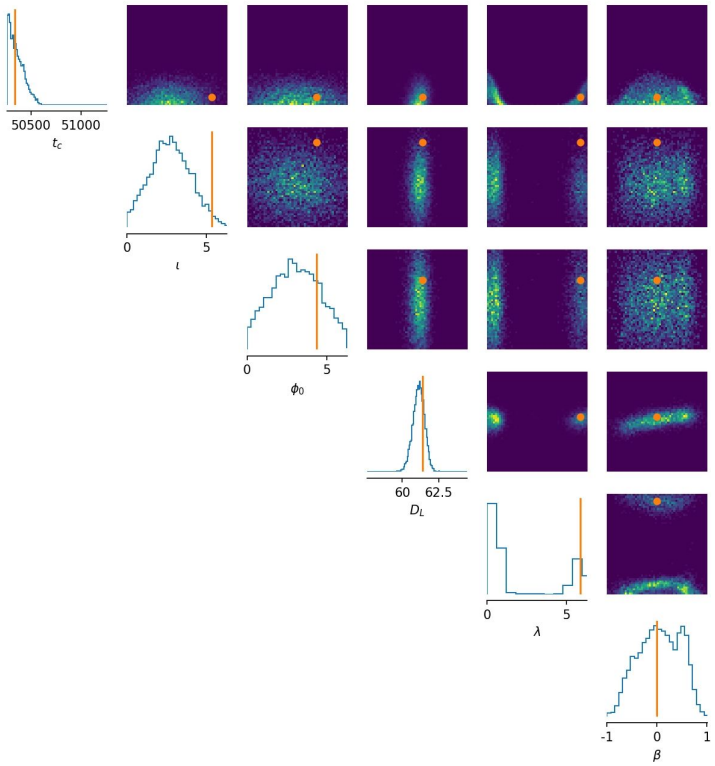
Method used to refine the posterior estimation at the sampling step, without touching the weights of the trained model.

- **Idea:** Compare the "proposal" distribution $q_\phi(\theta|\mathbf{x}_o)$ to the "target" distribution $p(\theta|\mathbf{x})$
- Our LISA response enables the computation of the real posterior $p(\theta|\mathbf{x})$ with the likelihood and prior distributions
- Each sample of the "proposal" distribution will be attached to a weight w_i defined as: $w_i = \frac{p(\theta_i|\mathbf{x}_i)}{q_\phi(\theta_i|\mathbf{x}_i)}$
- Small improvement of the method, doesn't change significantly the results

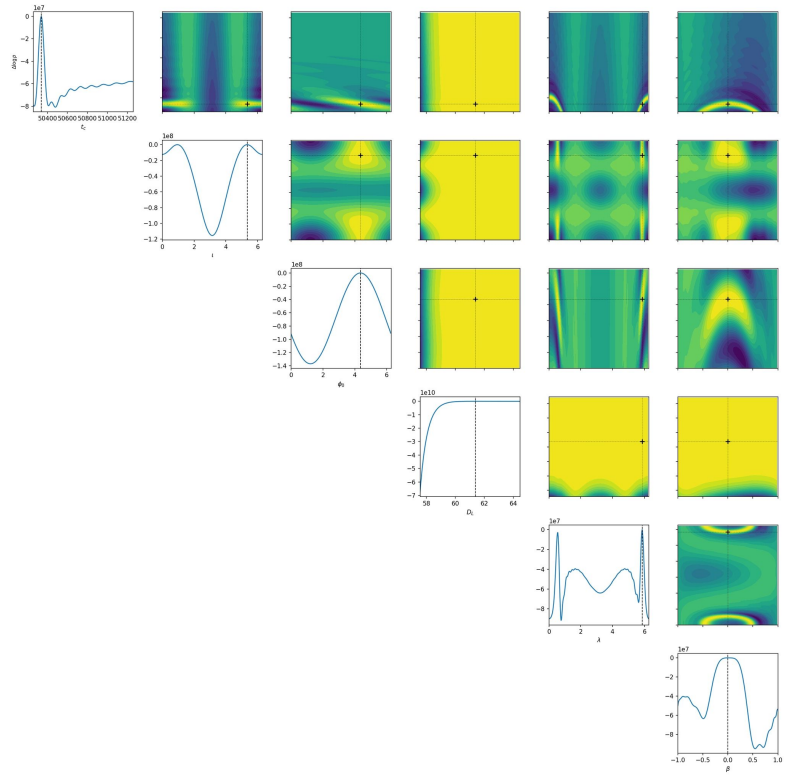
Preliminary sampling results

Cornerplots of posterior samples:

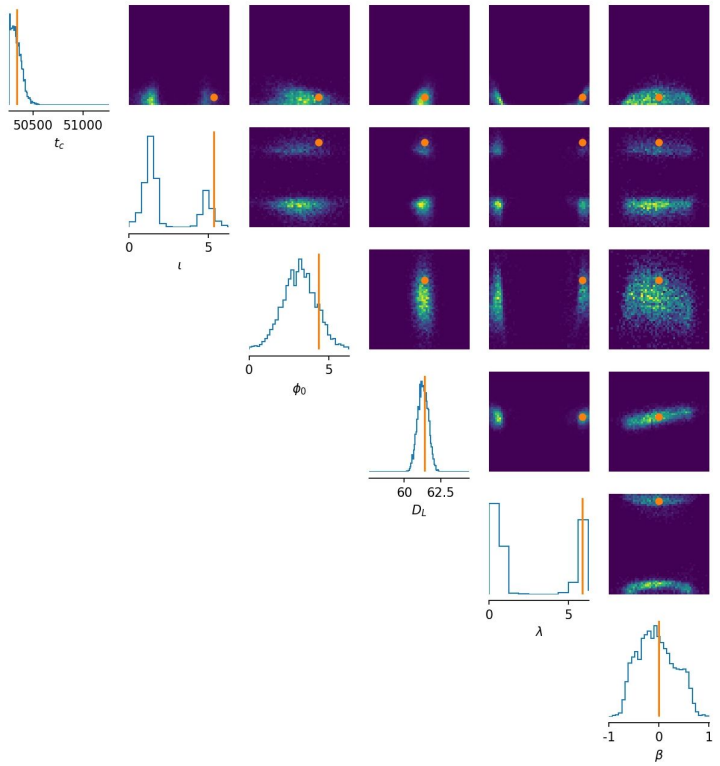
Example of a model trained on 100k simulations of 18000 time samples with a coalescence time window range of 1000 seconds, fixed masses and noise.



Posterior samples without importance sampling



Real 1D and 2D posterior marginals

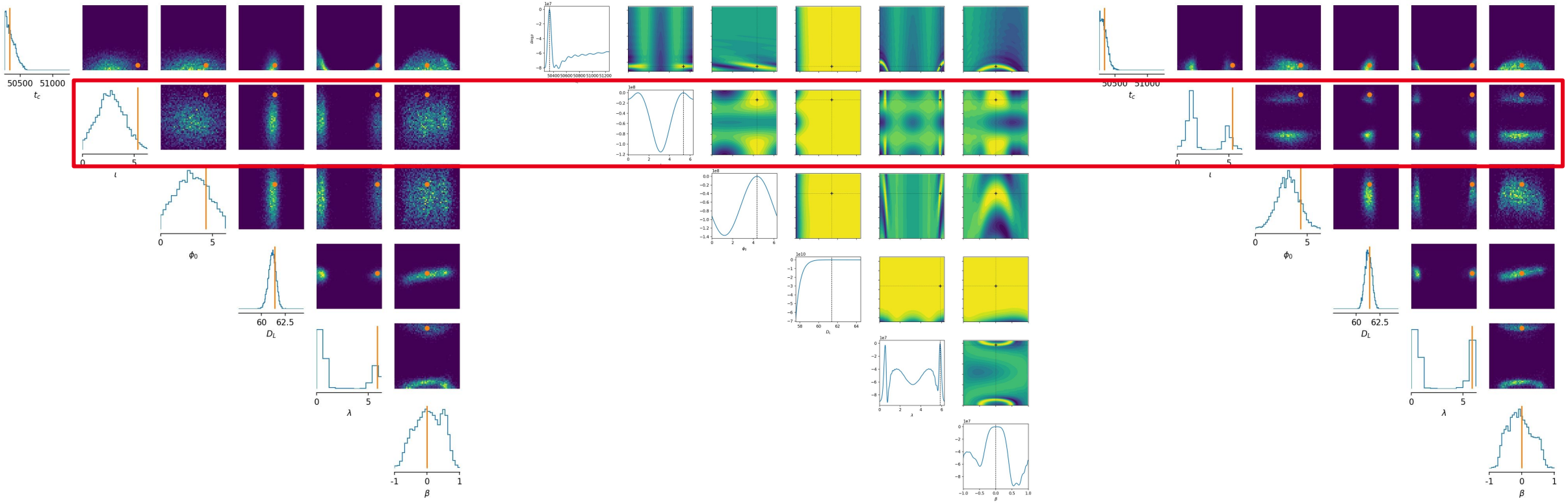


Posterior samples with importance sampling

Preliminary sampling results

Cornerplots of posterior samples:

Example of a model trained on 100k simulations of 18000 time samples with a coalescence time window range of 1000 seconds, fixed masses and noise.



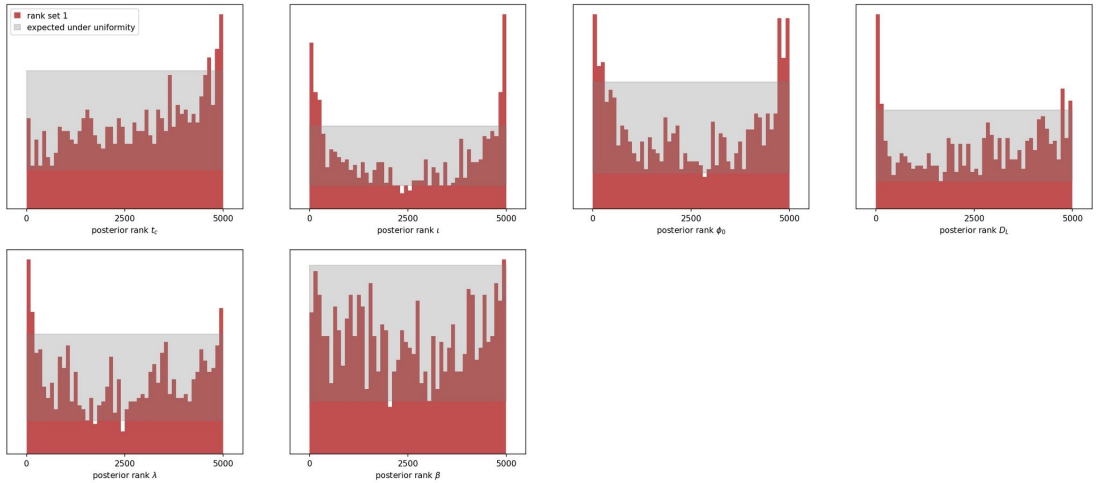
Posterior samples without importance sampling

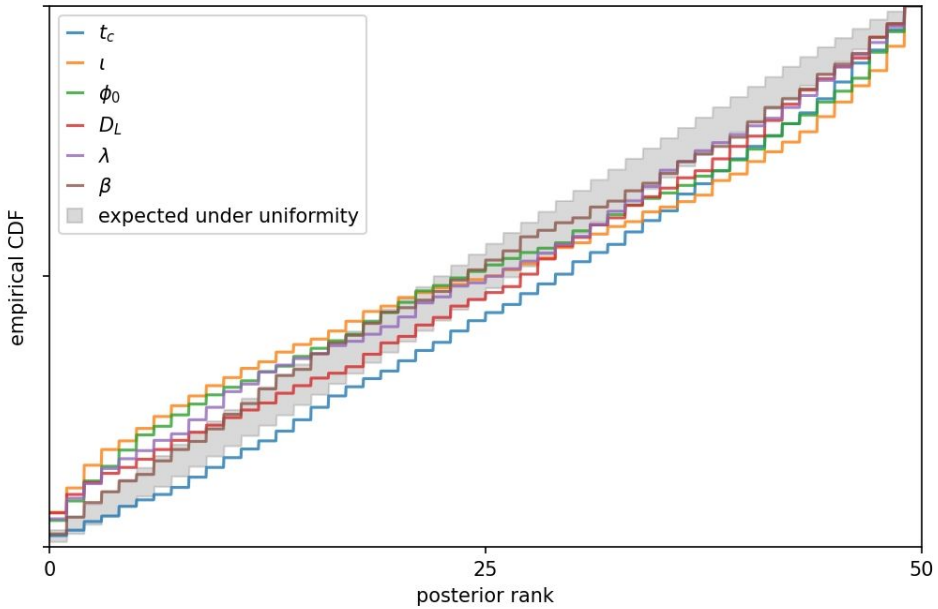
Likelihood surface on all the parameter space

Posterior samples with importance sampling

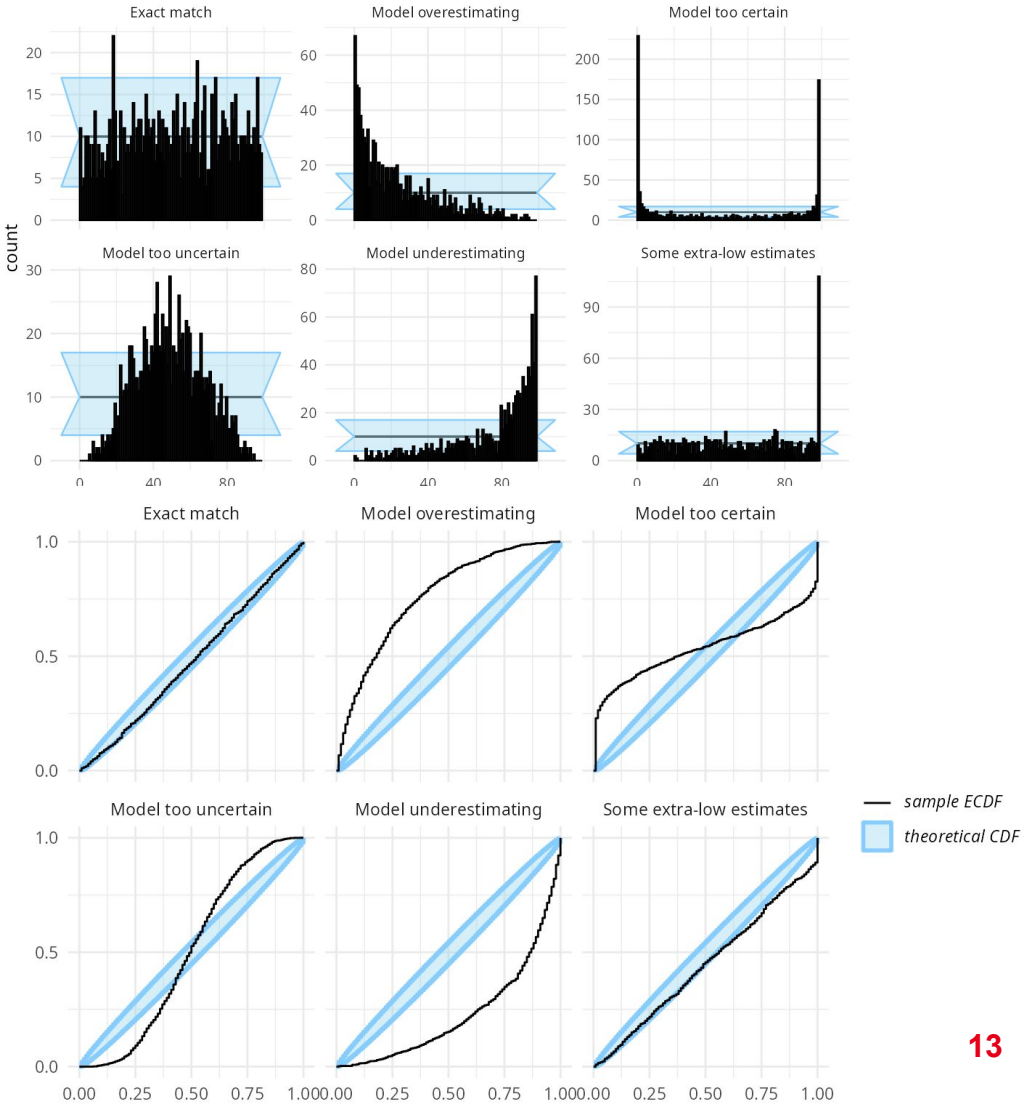
Validation of the training : Simulation based calibration (SBC)

Obtained SBC plots:





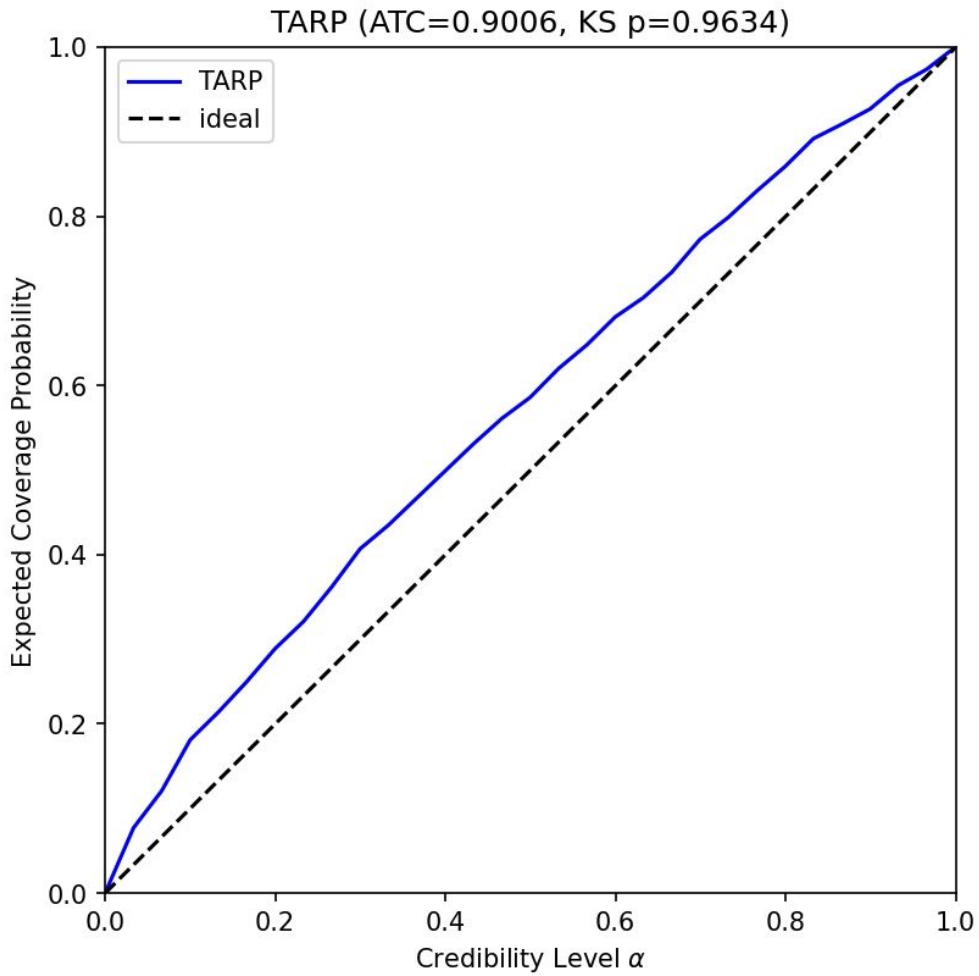
How to analyse SBC plots:



p-p plot calculation with TARP method

TARP method for p-p plots:

- Efficient method for p-p plot calculation
- Gives an aggregated p-p plot on the parameters
  - Plot the two cumulative distribution functions against each other
  - **If the model is well calibrated, the data will appear to be nearly a straight line**
- TARP gives a necessary and sufficient condition for posterior accuracy
- Detects inaccurate posterior estimators



Conclusion and further improvements

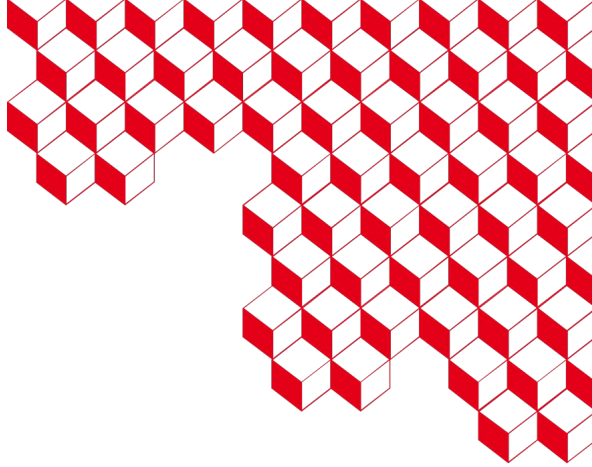
Results of the SBI pipeline:

- Simulator using an accelerated JAX response
- Dataset generation scripts using the simulator to create paired datasets $\mathscr{D} = \{(\mathbf{x}_i, \theta_i)\}_{i=1,\dots,N}$ according to our needs in data
- Scripts performing training and parallelizing it on GPUs
- Validation script performing importance sampling and calibration plot to diagnose biases in the trained model

Next steps:

For now we used SBI as the simplest way we could (simple summary statistics and default density estimator). The pipeline now makes tests easier for :

- Works on new summary statistics and embedding neural networks (Dictionary learning for time series, Transformer, STFT)
- Tests of other density estimators
- Investigate other parametrization
- Apply on Mojito data
- Use more realistic waveforms



Thank you for your attention.

LISA team @ IRFU, CEA-Saclay

Extra slide: Simplified waveform

$$h_+ = h_{\theta\theta} = -2\left(1+\cos^2\theta\right)\frac{\mu}{R}(M\omega)^{2/3}\cos\left[2\omega(t-R)-\phi_0\right]$$

$$h_\times = h_{\theta\phi} = -4\cos\theta\,\frac{\mu}{R}(M\omega)^{2/3}\sin\left[2\omega(t-R)-\phi_0\right].$$

Ref: Babak (2020), “Gravitational Waves from Coalescing Binaries”