# NMMA long form update: jax

Thibeau Wouters

February 20, 2024

# Table of Contents

# Table of Contents

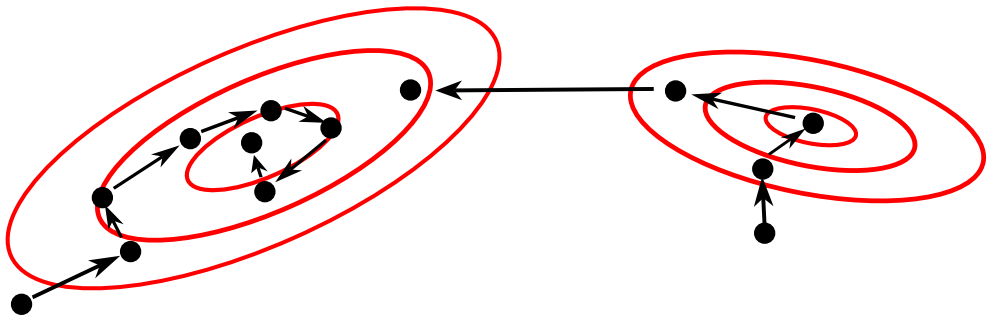
# Parameter estimation

- Parameter estimation (PE): get posterior of GW/EM parameters $\theta$

$$p(\theta|d) = \frac{p(d|\theta)p(\theta)}{p(d)}$$

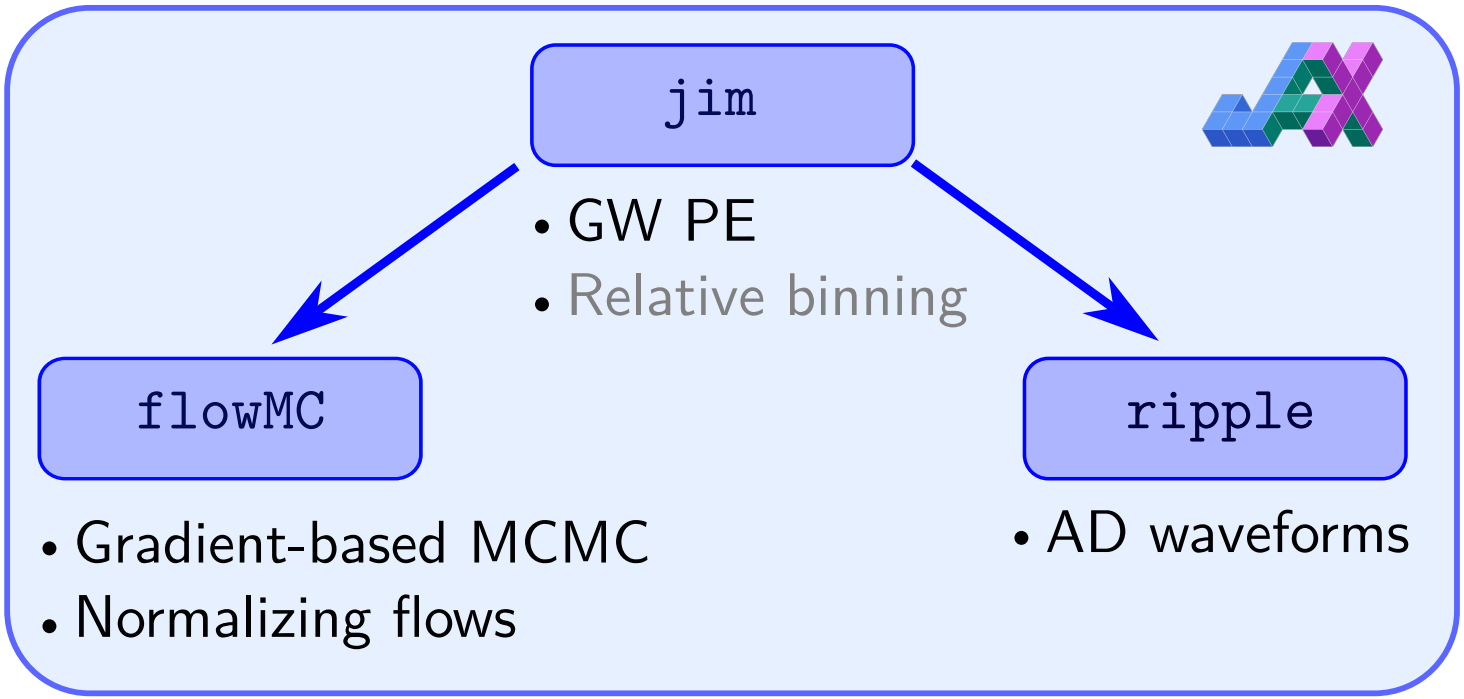
- Sampling via Markov Chain Monte Carlo (MCMC) [1]

How to sample from high-dimensional, multi-modal posteriors?



# Overview

1. `jax` [2]
2. `flowMC`: Normalizing flow-enhanced, gradient-based MCMC [3, 4]
3. `ripple`: Automatically-differentiable (AD) GW [5]
4. `jim`: Accelerated PE for GW [6]



# Table of Contents

# Why `jax`?

What are the benefits of `jax` for MCMC?

1. Automatic differentiation (AD)
2. Just-in-time (JIT) compilation
3. GPU acceleration
4. Parallelization

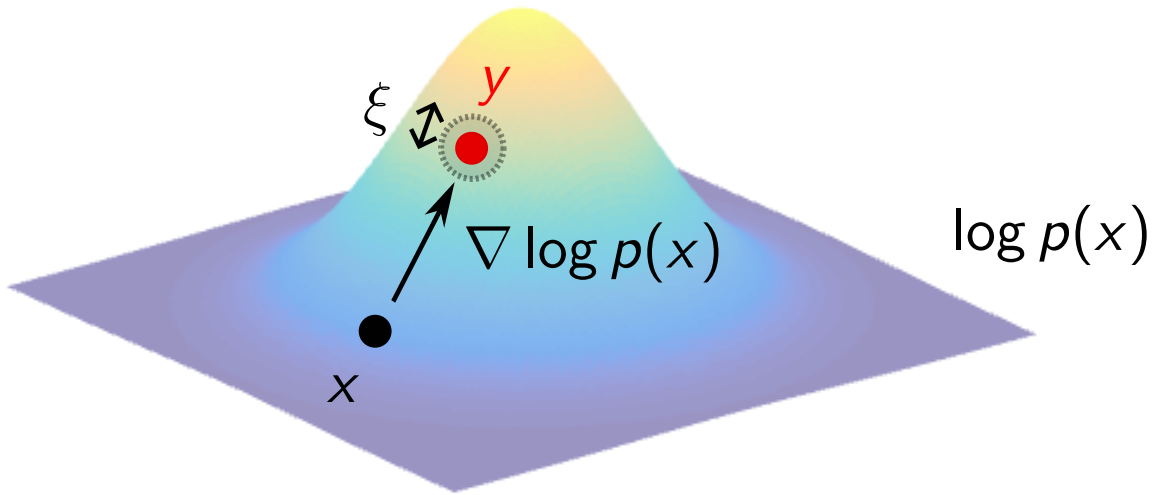
# Table of Contents

# flowMC – local sampling

1. **Local sampling**: MALA (Metropolis-adjusted Langevin algorithm)
   - Proposal $y$: Langevin diffusion

$$y = x + \frac{\epsilon^2}{2} \nabla \log p(x) + \epsilon \xi$$

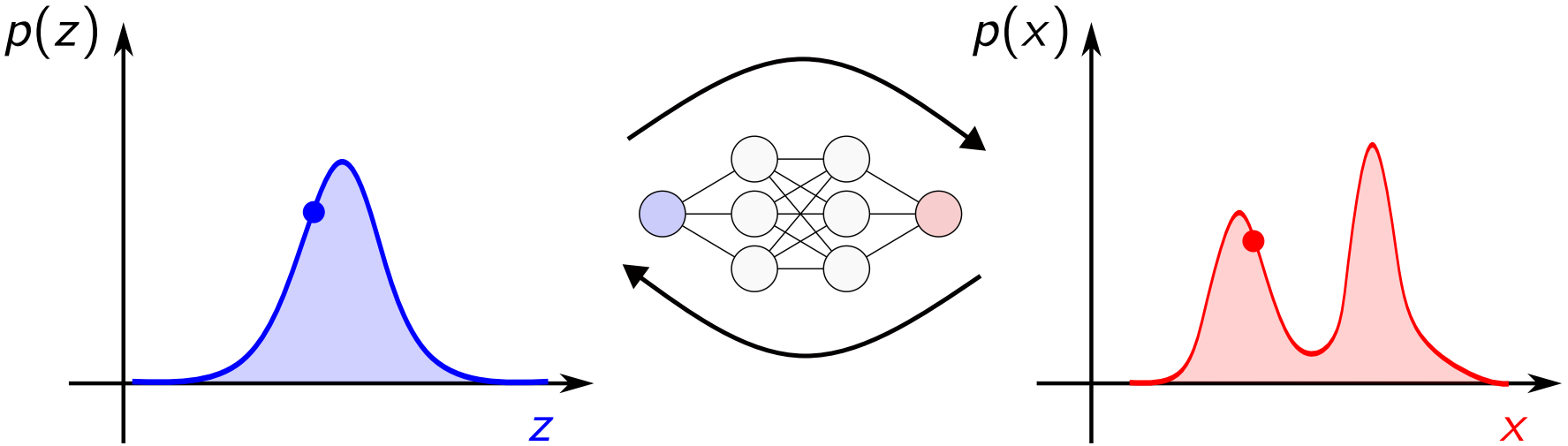
   - Metropolis-Hastings acceptance step



# flowMC – normalizing flows

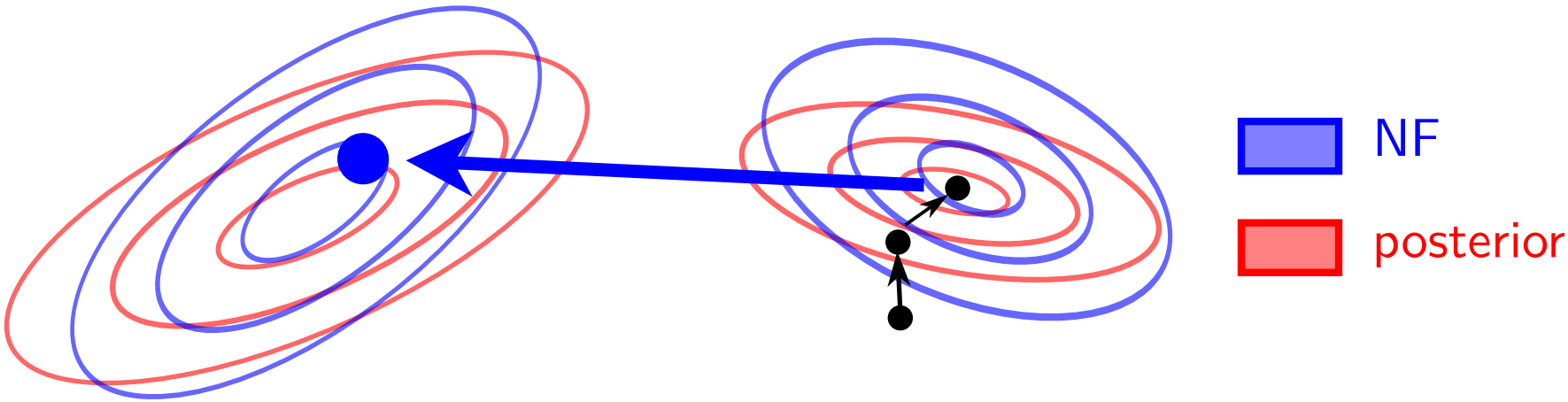
Normalizing flows (NF):

- Latent space: easy to sample (e.g. Gaussian)
- Data space: distribution learned from samples
- Enable approximate sampling from complicated distributions



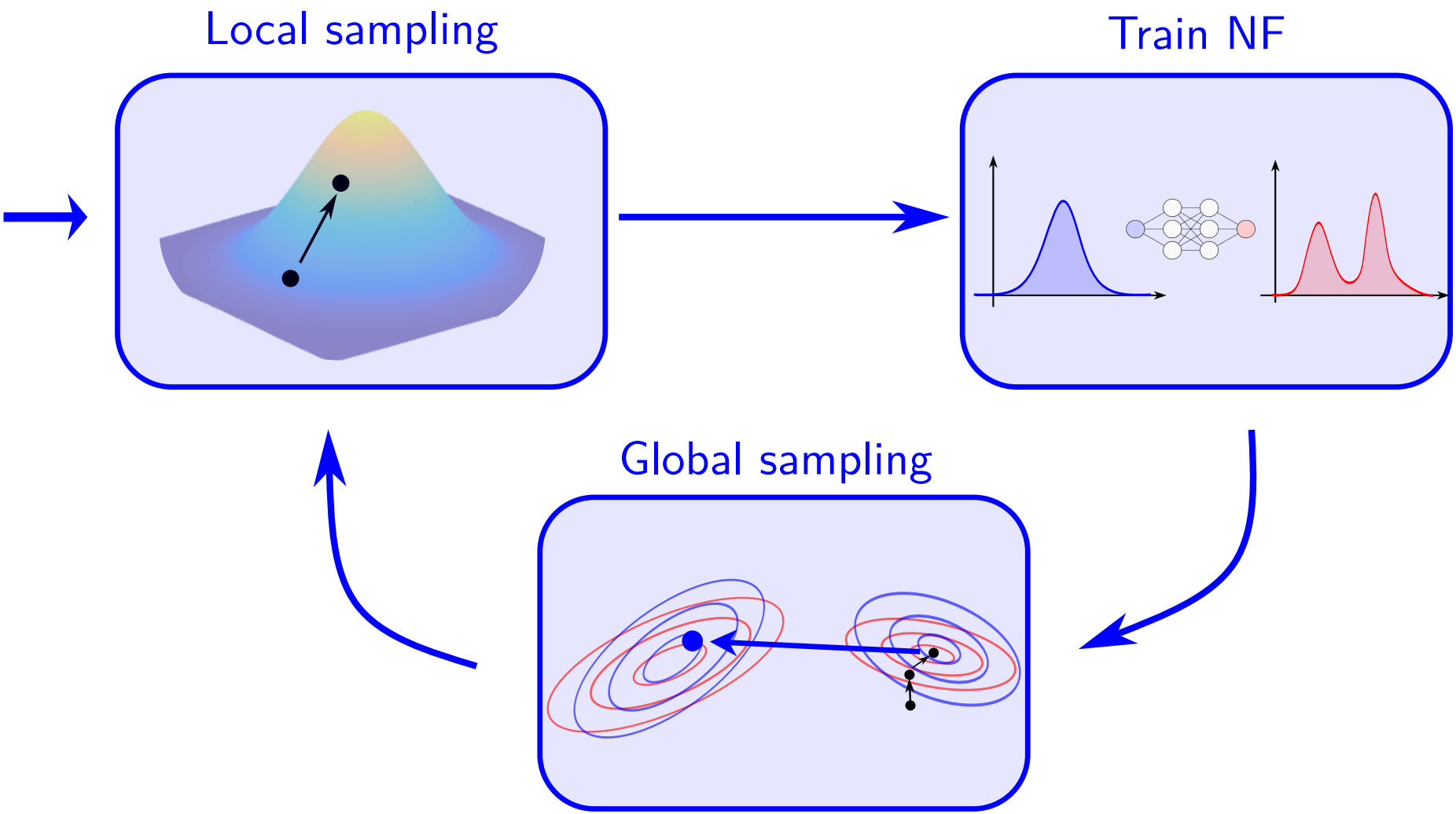
## 2 Global sampling

- Global proposal by sampling from NF
- Metropolis-Hastings acceptance step



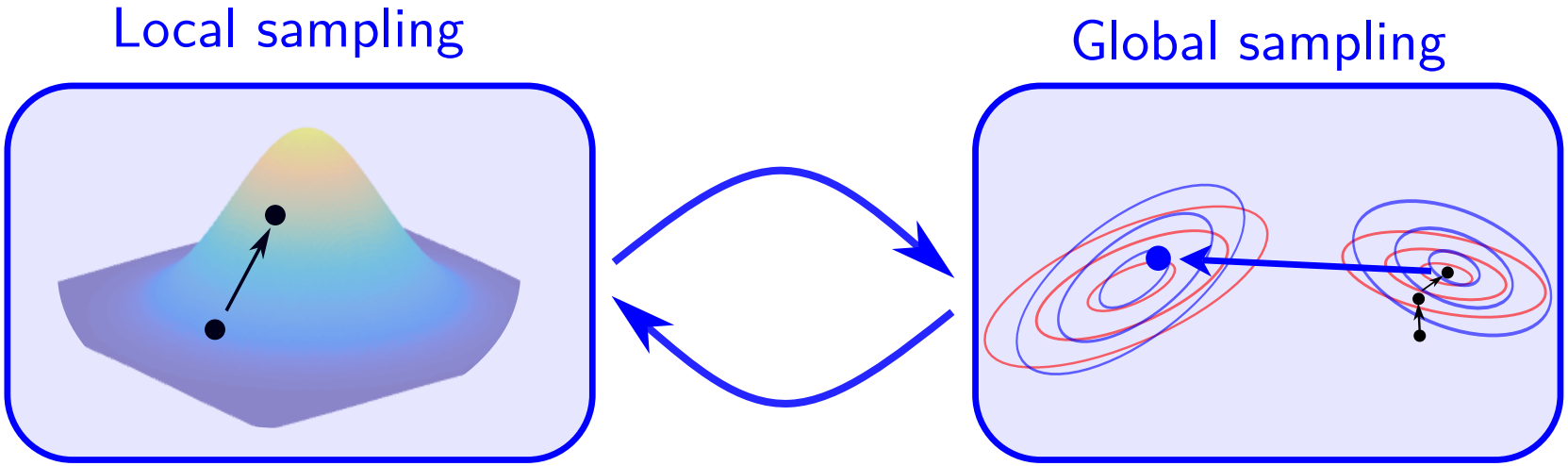
# flowMC – complete algorithm

**Training loop** & Production loop



# flowMC – complete algorithm

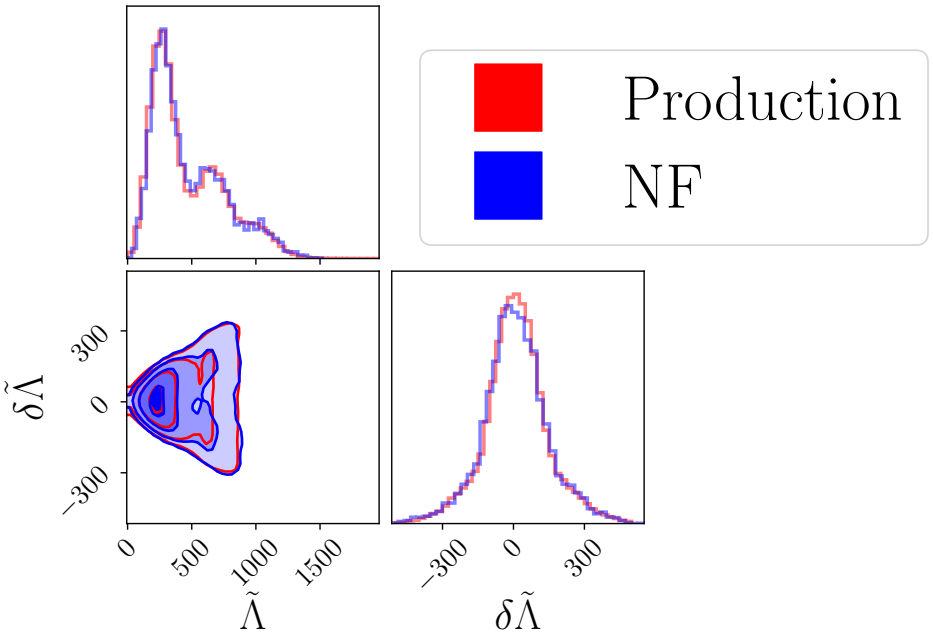
Training loop & **Production loop**



# Table of Contents

# Results – GW

- Reproduced PE for GW170817 & GW190425 with TaylorF2
- Wrapping up injection studies
- Runtime: ~~30 min – 1 hour~~ $(\mathbf{17.11 \pm 2.65})$ **min** (34 runs)
- **Current work**: tuning robustness/performance



# Results – EM

- Implemented surrogate model of Bu2022Ye with `flax` [7]
- Started on incorporating a `jax`-compatible likelihood for EM, but gives NaNs
- **Future work**: debug likelihood function, run first KN PE with `jax`

# Table of Contents

# Demo

- Time for a demo!
- Interested? Homework: check the Google docs

# References


[1] Steve Brooks et al. *Handbook of markov chain monte carlo*. CRC press, 2011.

[2] James Bradbury et al. *JAX: composable transformations of Python+NumPy programs*. Version 0.3.13. 2018. URL: http://github.com/google/jax.

[3] Marylou Gabrié, Grant M Rotskoff, and Eric Vanden-Eijnden. "Efficient bayesian sampling using normalizing flows to assist markov chain monte carlo methods". In: *arXiv preprint arXiv:2107.08001* (2021).

[4] Kaze WK Wong, Marylou Gabrié, and Daniel Foreman-Mackey. "flowMC: Normalizing-flow enhanced sampling package for probabilistic inference in Jax". In: *arXiv preprint arXiv:2211.06397* (2022).

[5] Thomas DP Edwards et al. "ripple: Differentiable and Hardware-Accelerated Waveforms for Gravitational Wave Data Analysis". In: *arXiv preprint arXiv:2302.05329* (2023).

[6] Kaze WK Wong, Maximiliano Isi, and Thomas DP Edwards. "Fast gravitational wave parameter estimation without compromises". In: *arXiv preprint arXiv:2302.05333* (2023).

[7] Jonathan Heek et al. *Flax: A neural network library and ecosystem for JAX*. Version 0.8.1. 2023. URL: http://github.com/google/flax.