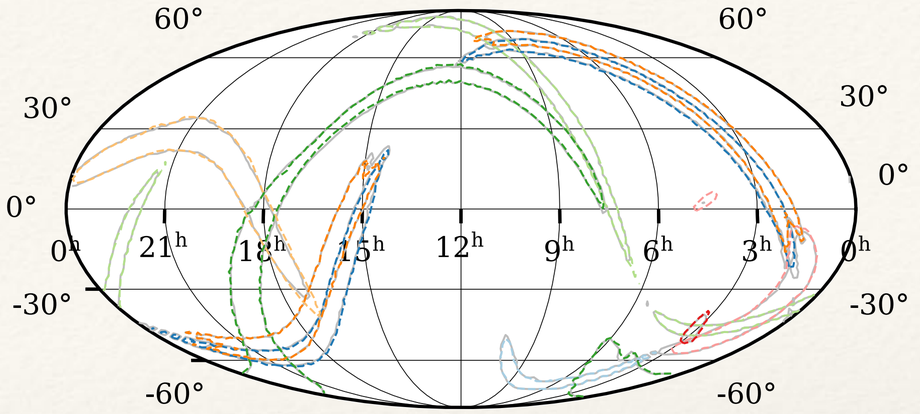


Gravity Seminar, Niels Bohr Institute — 14 February 2024

Simulation-Based Inference for GW Parameter Estimation

Stephen Green

University of Nottingham
UK | CHINA | MALAYSIA

with

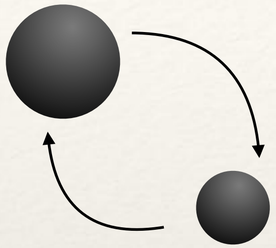
Maximilian Dax
Jonathan Gair
Jonas Wildberger
and others

Outline

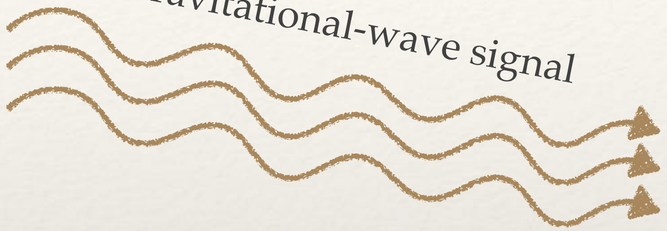
- Classical methods for gravitational wave parameter estimation
- Neural density estimation and simulation-based inference
- Validating results with importance sampling
- Next steps: Flow matching and population inference

Introduction

Merging black hole and/or neutron star binaries

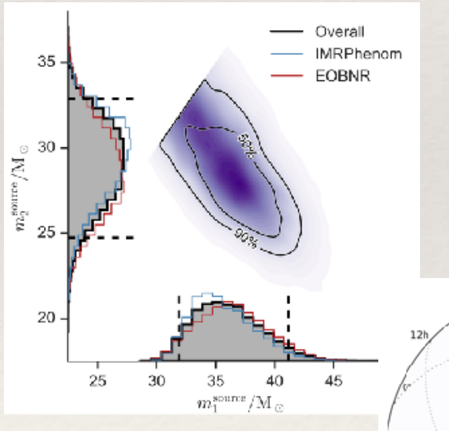


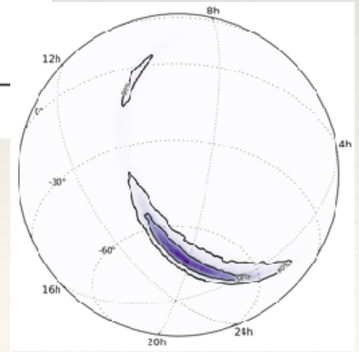
Gravitational-wave signal

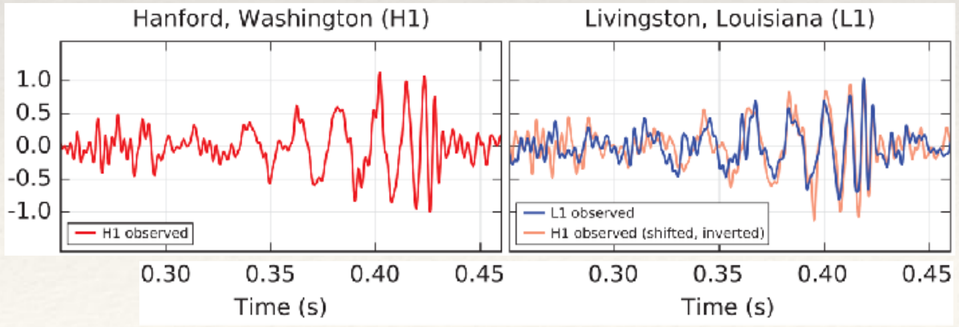


LIGO and Virgo observatories

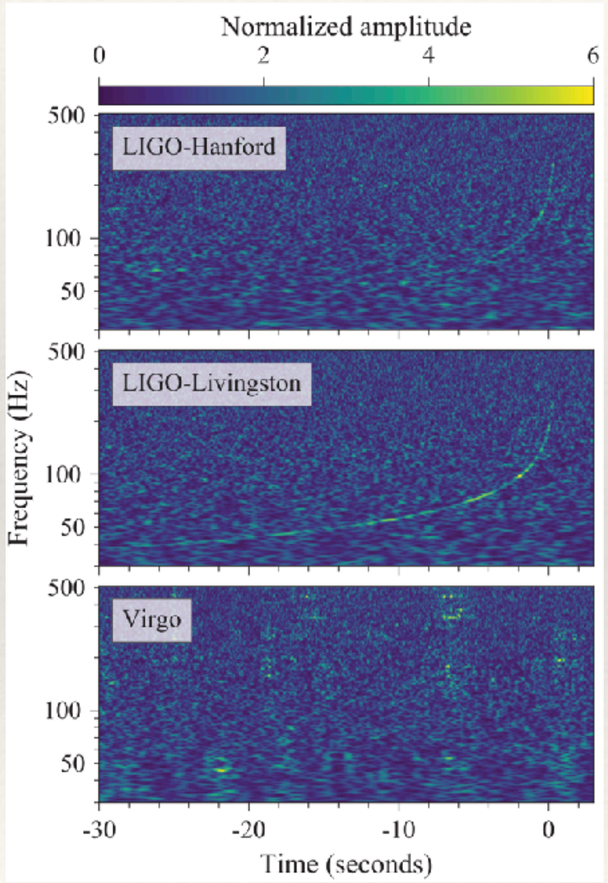
Theoretical models + Bayesian inference





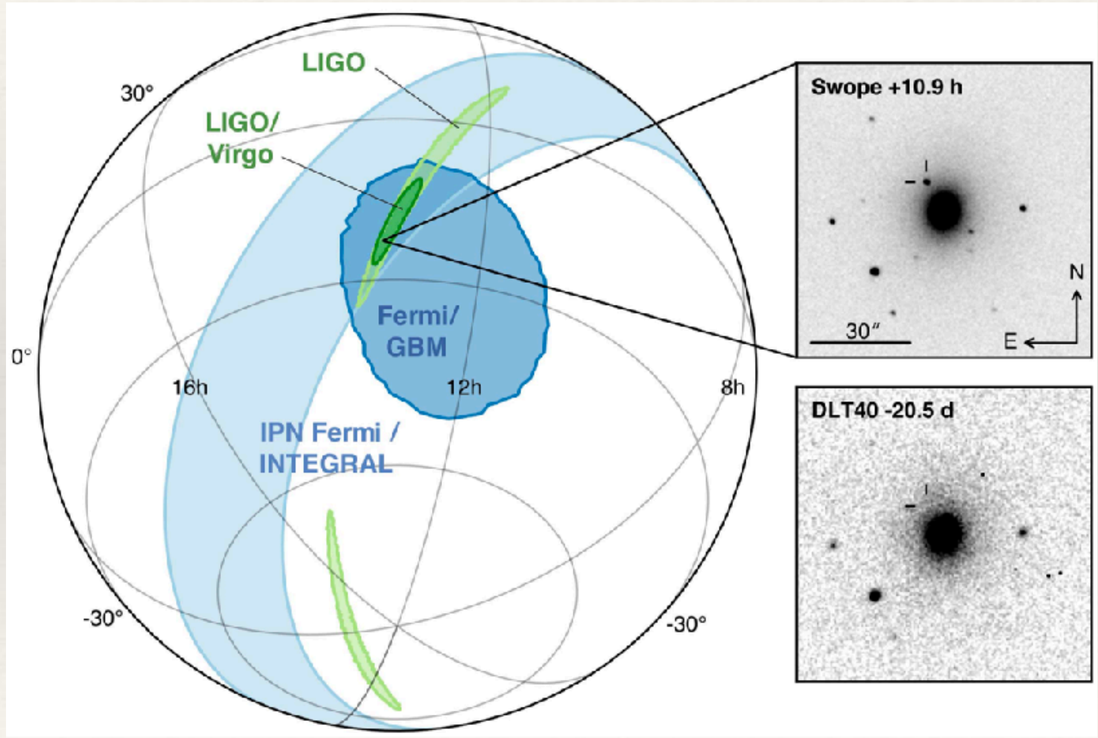


Binary neutron star mergers



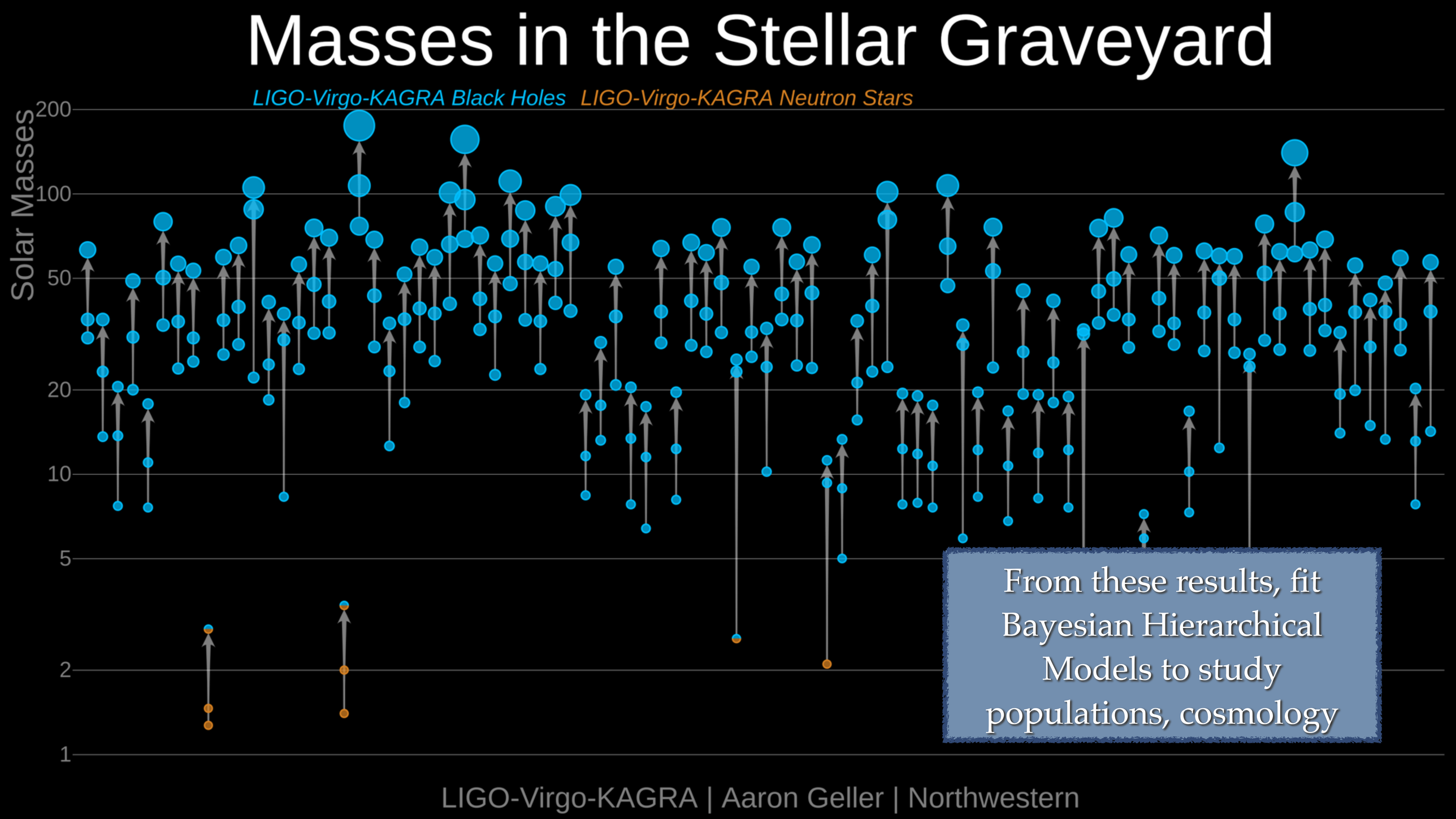
PRL 119, 161101 (2017)

- GW170817: Sky localization enables multimessenger astrophysics

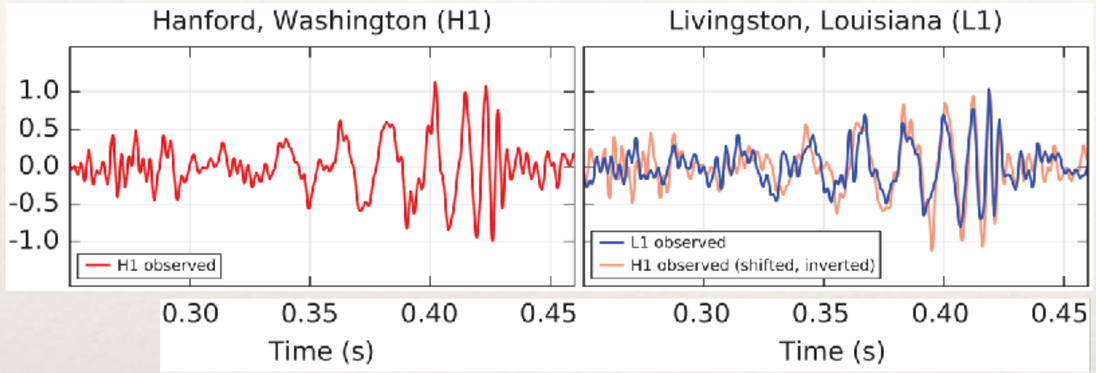


ApJ Lett 848:L12 (2017)

Masses in the Stellar Graveyard

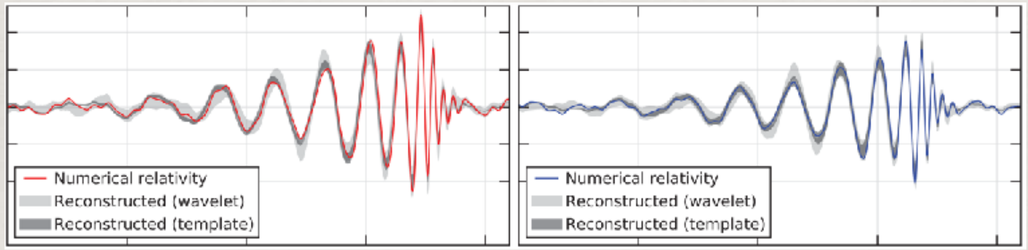


Basics of GW parameter estimation



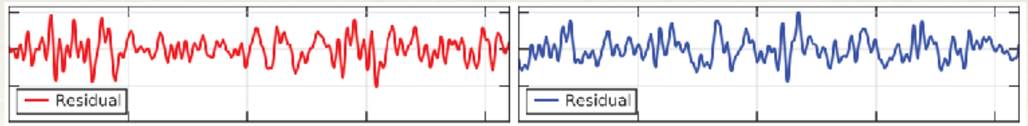
observed data d

$=$

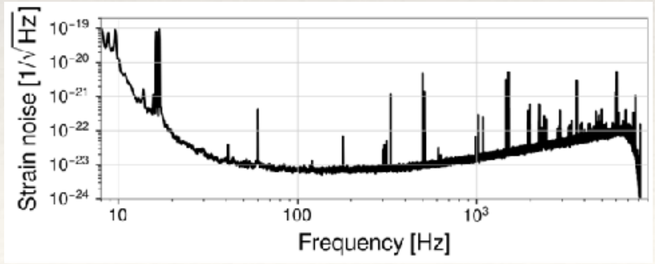


signal $h(\theta)$

$+$



noise



Bayesian inference

Posterior

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

Likelihood assumes **stationary Gaussian** detector noise

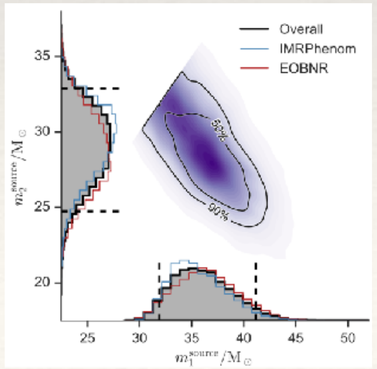
$$p(d\,|\,\theta) = \mathcal{N}(h_I(\theta), S_{\mathrm{n},I})$$

waveform model | power spectral density

Prior

e.g., uniform in masses, spins, sky position, …

- Finally, draw samples



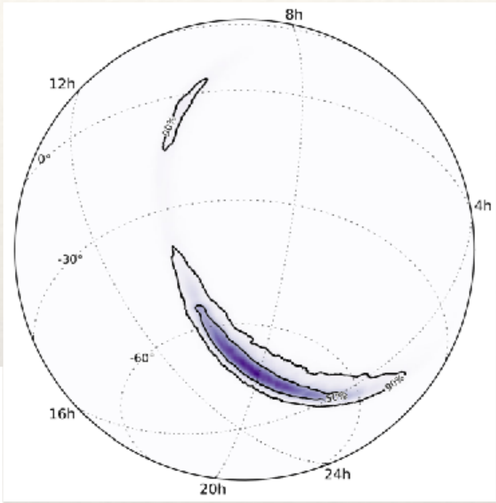
Sampling

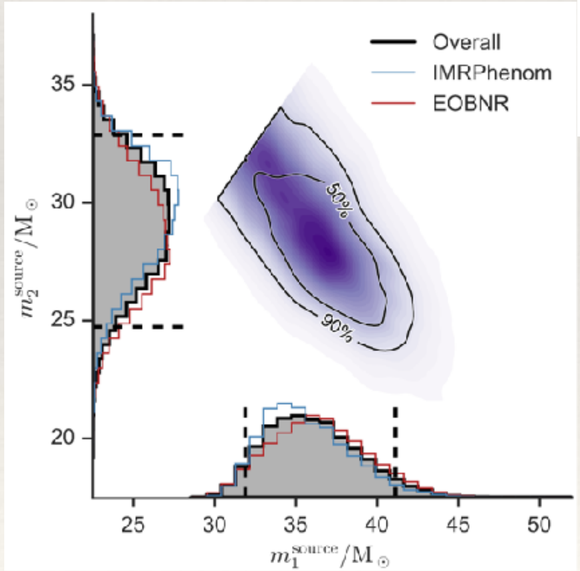
- **Stochastic:** Markov chain Monte Carlo (MCMC) or nested sampling

$$\theta \sim p(\theta \mid d)$$

  - **Explore** parameter space
  - **Generate** waveforms
  - **Compare** to data

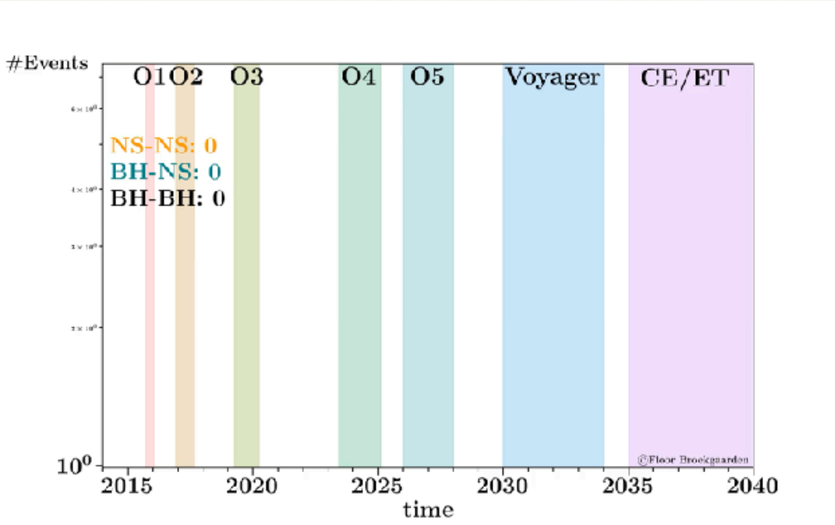
- Requires **millions** of likelihood evaluations per detection.





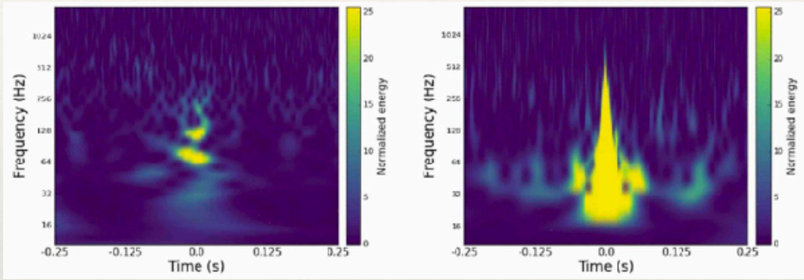
Why explore new methods?

- Current analyses based on MCMC / nested sampling are expensive
- Huge numbers of events expected in the future

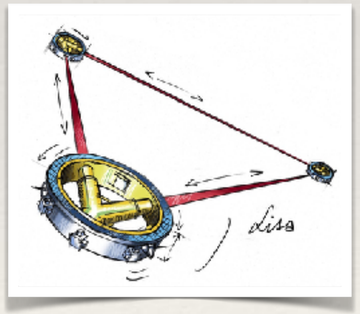


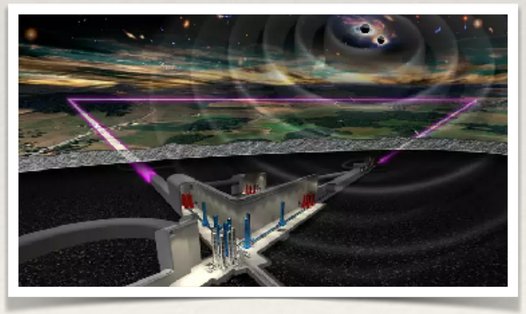
Movie credit:
Floor Broekgaarden

- Forced to make approximations, e.g., that noise is stationary and Gaussian



- Future data will be far more complex





Simulation-based inference + deep learning can solve all of these problems

Neural posterior estimation

- Learn a neural network representation of the posterior

$$q_\phi(\theta \mid d) \approx p(\theta \mid d)$$

millions of tunable parameters

- Enable fast sampling for any data d.

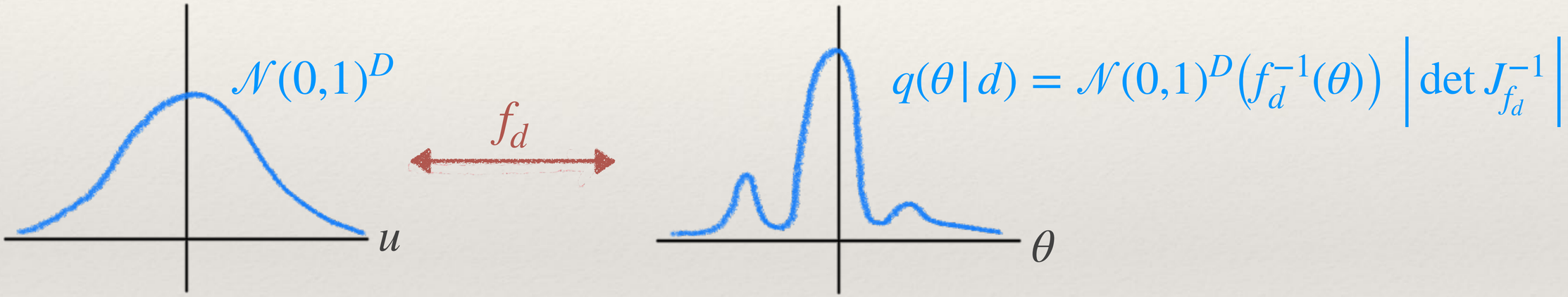
Neural posterior estimation

$$q_\phi(\theta \mid d) \approx p(\theta \mid d)$$

- **Questions**
  - How do we represent a conditional distribution using neural networks?
  - What do we train on?
  - How do we know our answer is right?

Normalizing flow

- Represent complex distribution in terms of a **mapping** $f_d : u \mapsto \theta$ from simpler distribution:

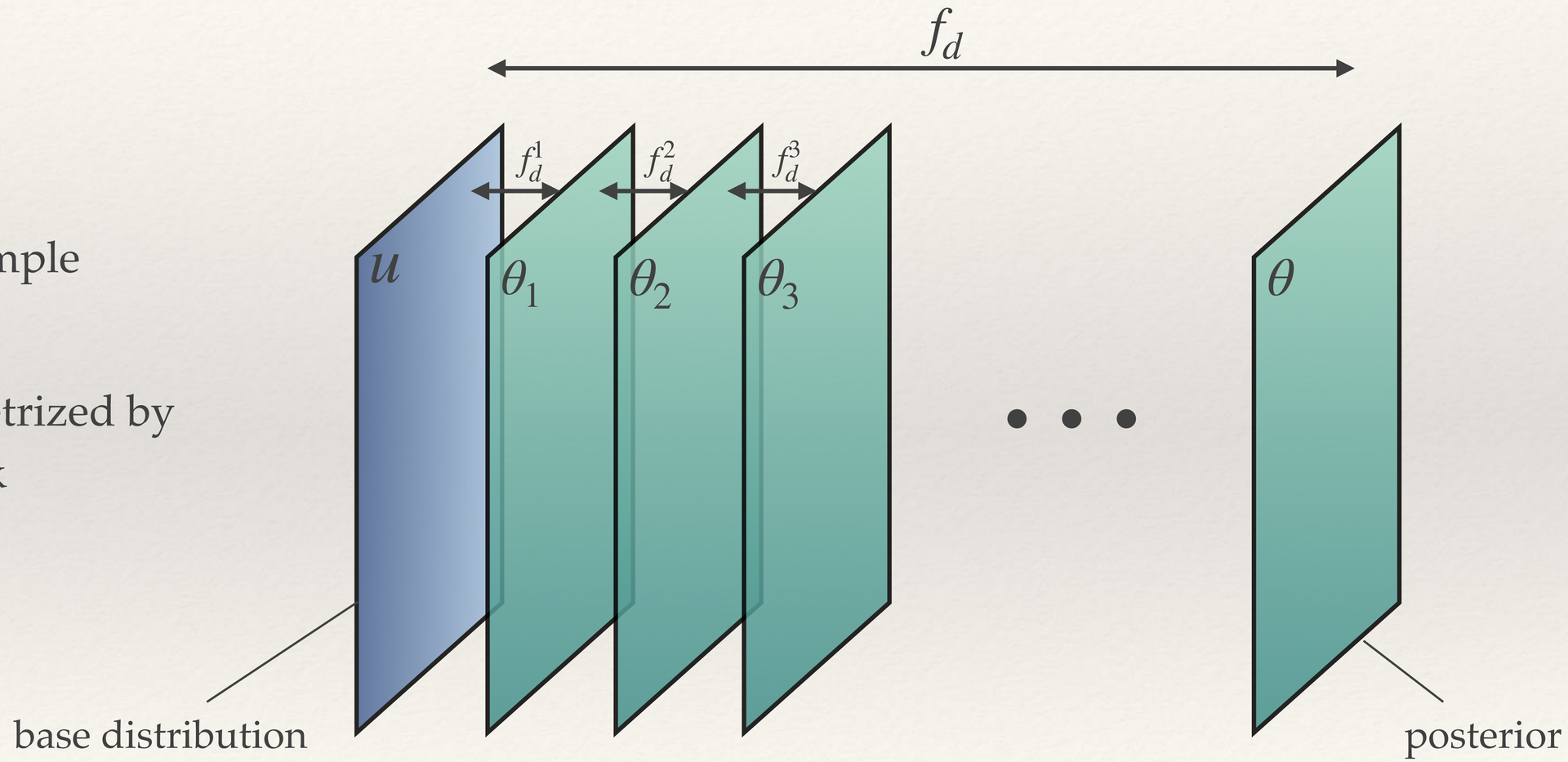


- **Properties of** f_d
  1. invertible
  2. simple Jacobian determinant

→ $q(\theta \mid d)$ has **fast sampling and density evaluation**

Normalizing flow

- Sequence of simple transforms
- Each f_d^i parametrized by neural network



Normalizing flow

- **Requirements:**
  1. Invertible ✓
  2. Simple Jacobian determinant ✓

$$\det J_{f_d} = \prod_{i=\frac{D}{2}+1}^{D} c_i' \left(u_i; u_{1:\frac{D}{2}}, d\right)$$

- Use a sequence of "coupling transforms":

$$f_d^i(u) = \begin{cases} u_i & \text{if } i \leq D/2 \\ c_i \left(u_i; u_{1:\frac{D}{2}}, d\right) & \text{if } i > D/2 \end{cases}$$

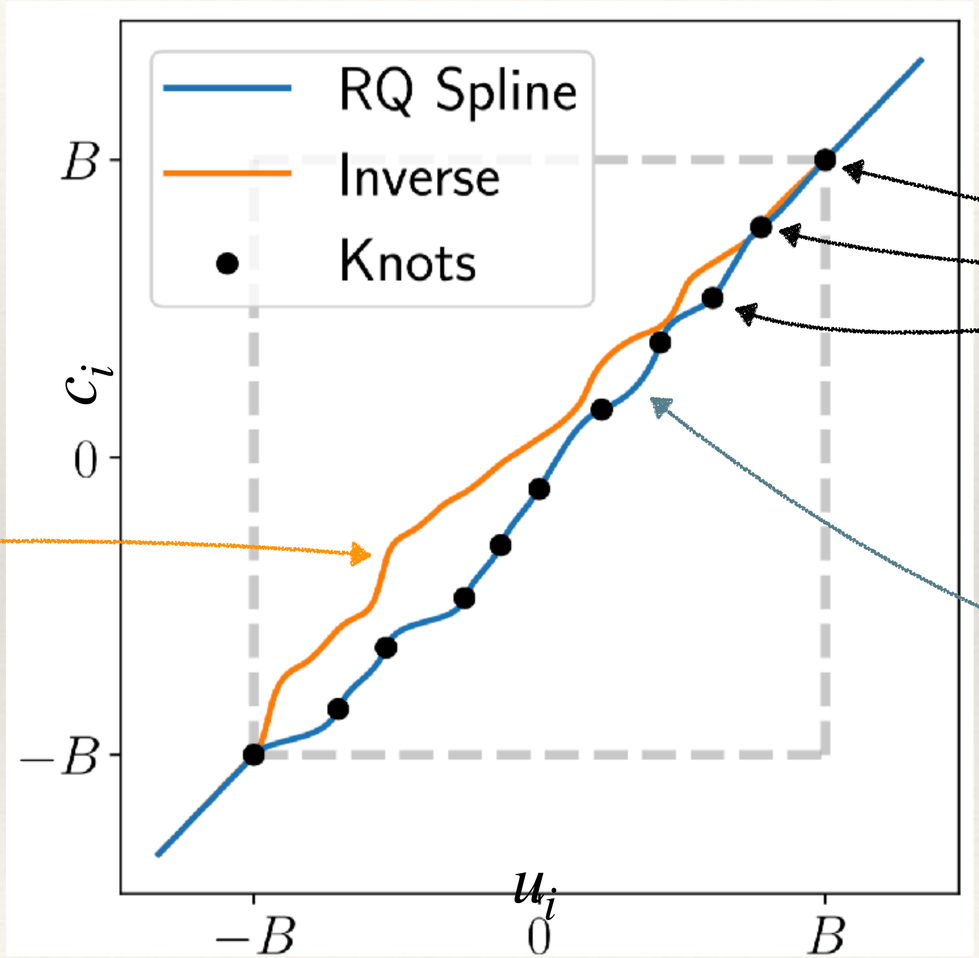
Hold fixed half of the components

Transform remaining components element-wise, conditional on other half and s.

- c_i should be **differentiable** and have **analytic inverse** with respect to u_i. Use a spline (Durkan+ 2019)

Normalizing flow

- **Spline flow**
  (Durkan et al, 2019):



knots and **derivatives** output of neural network; input $(u_{1:\frac{D}{2}}, d)$

analytic inverse

rational-quadratic spline interpolation

Figure: Durkan *et al* (2019)

Training

Train $q_\phi(\theta \mid d) \rightarrow p(\theta \mid d)$

1. Specify a "loss function" $L[q_\phi]$.
   - Depends on a set of training examples $\{(\theta^{(i)}, d^{(i)})\}$
2. Minimize the loss
   - Set $\phi_0 = \arg\min_\phi L[q_\phi]$ by following gradients $\partial_\phi L[q_\phi]$

Naively require posterior samples to model $p(\theta \mid d)$

Very expensive!

Training

$$
\begin{aligned}
L[q_\phi] &= \mathbb{E}_{p(d)} D_{\mathrm{KL}}(p \| q_\phi) \\
&= \int dd\, p(d) \int d\theta\, p(\theta|d) \log \frac{p(\theta|d)}{q_\phi(\theta|d)} \\
&\simeq \int d\theta\, p(\theta) \int dd\, p(d|\theta) \left[-\log q_\phi(\theta|d) \right] \\
&\approx \frac{1}{N} \sum_{\substack{\theta^{(i)} \sim p(\theta) \\ d^{(i)} \sim p(d|\theta^{(i)})}} -\log q_\phi(\theta^{(i)}|d^{(i)})
\end{aligned}
$$

Bayes' theorem

(1) sample $\theta^{(i)}$ from the prior,
(2) simulate $d^{(i)}$ = signal + noise.

No posterior samples.

No likelihood evaluations.

Changing PSDs

- In reality, **detector noise is not totally stationary**. Rather the spectrum $S_{\rm n}(f)$ varies from event to event.
- Account for this by augmenting the training to include a collection of PSDs, $S_{\rm n}(f) \rightarrow \{S_{\rm n}^{(i)}(f)\}$.
- To generate data,

1. Choose a PSD $\quad S_{\rm n}^{(i)} \sim p(S_{\rm n})$
2. Generate noise $\quad n^{(i)} \sim \mathcal{N}(0, S_{\rm n}^{(i)})$
3. Add a signal $\quad d^{(i)} = h(\theta^{(i)}) + n^{(i)}$

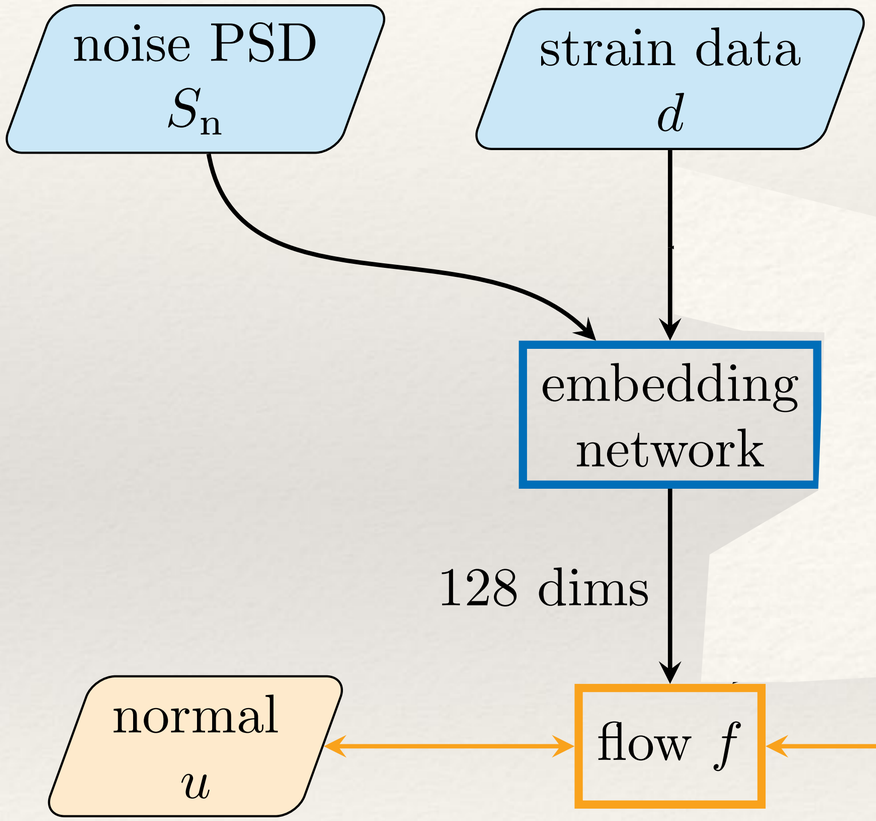
Condition the model also on the PSD

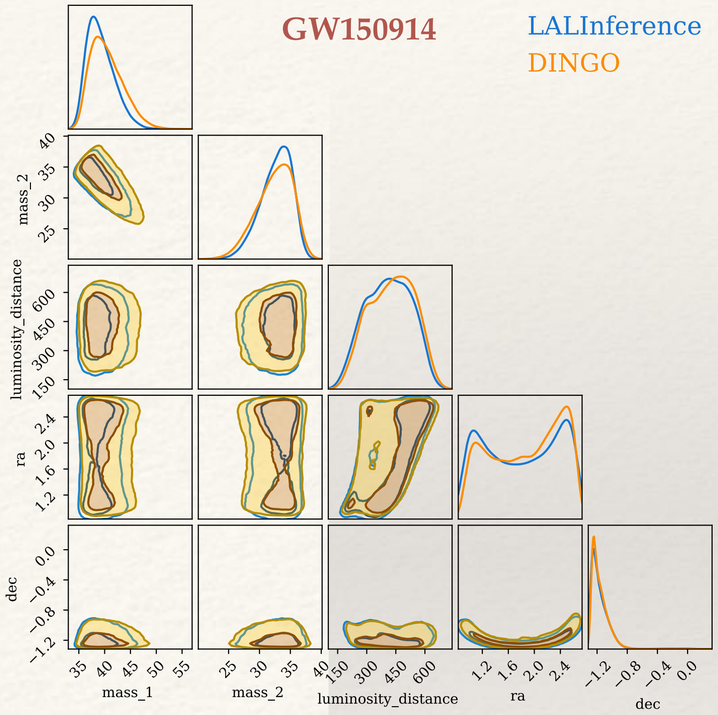
$$q_\phi(\theta \,|\, d) \rightarrow q_\phi(\theta \,|\, d, S_{\rm n})$$

Picture so far

Dax, SRG+ (PRL 2021)

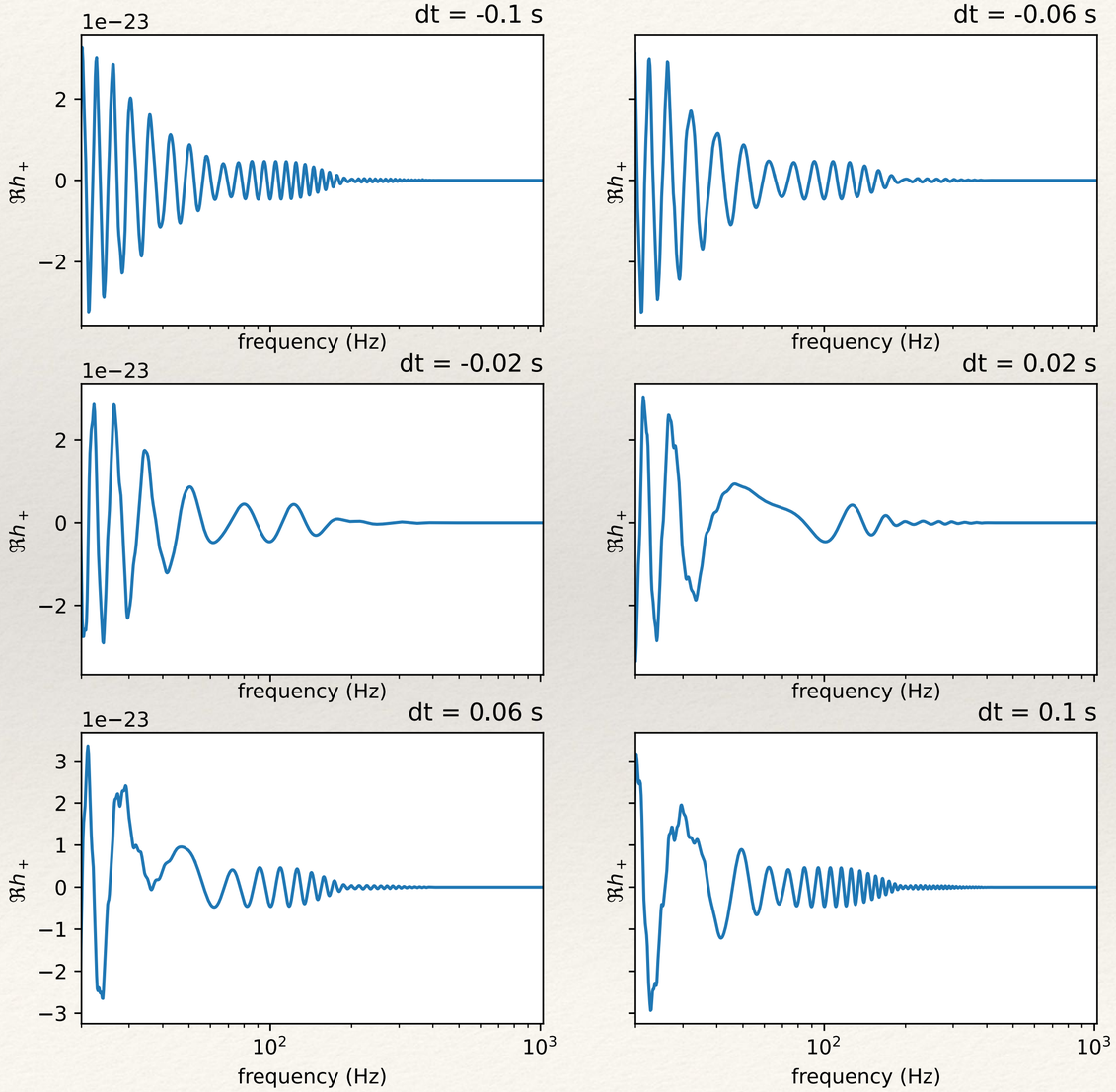
Dataset of training PSDs either **real** or **forecasted**



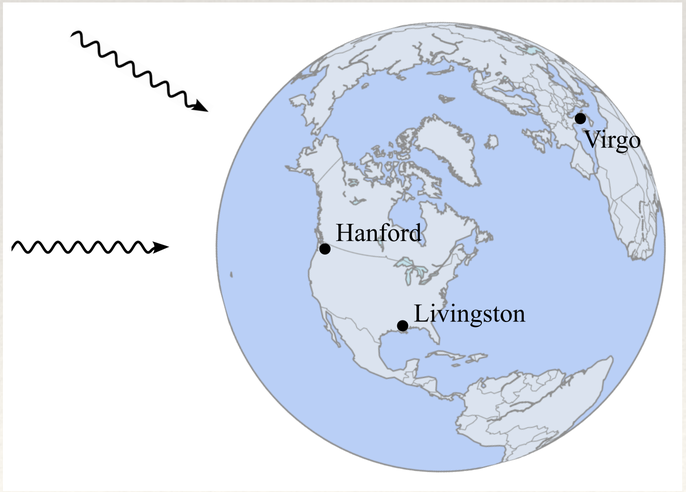


Simplify data using symmetries

Dax, SRG+ (ICLR, 2022)



- Frequency-domain data
  - Time shift corresponds to multiplication by $e^{-2\pi i f \delta t}$
- Variation in sky position + overall coalescence time
  - Time shifts in each detector $\delta t \approx 0.1$ s



Hard to learn!

Group-equivariant NPE

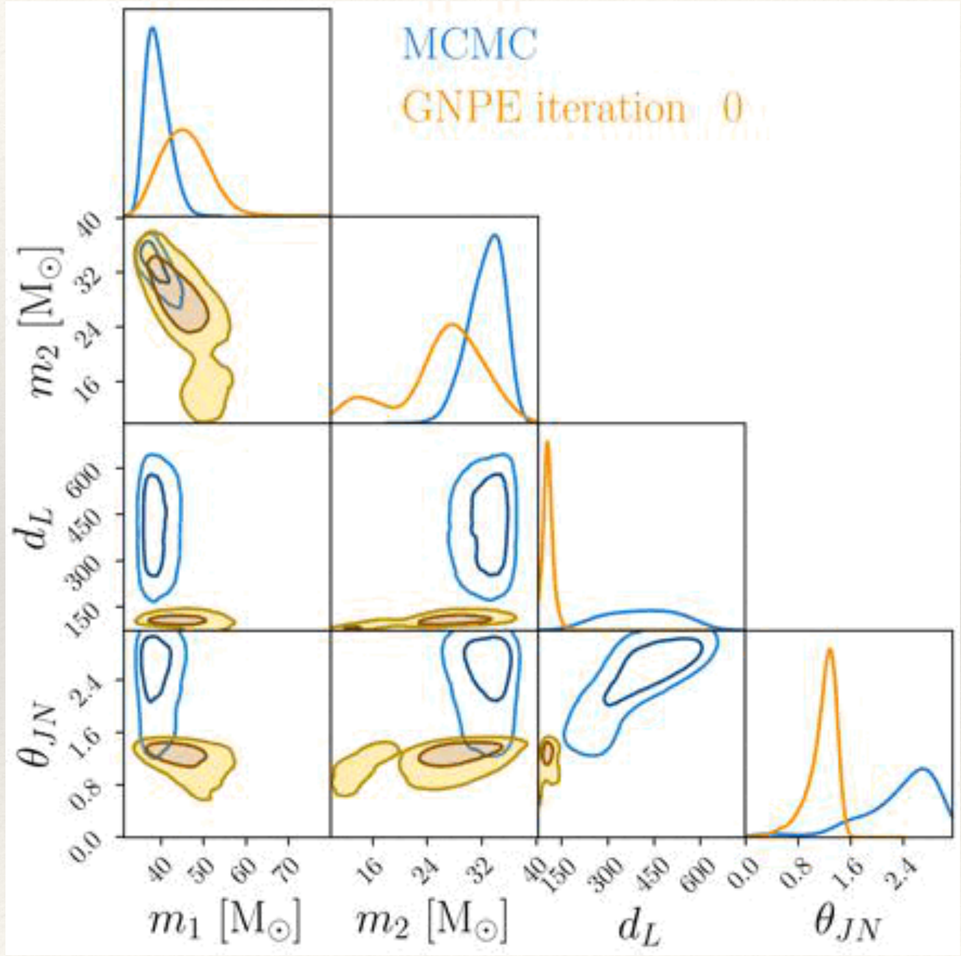
Dax, SRG+ (ICLR, 2022)

- Consider overall time translations. The true posterior is covariant under joint transformations of parameters and data,

$$p(t_c \,|\, d) = p(t_c + \delta t \,|\, d \cdot e^{-2\pi i f \delta t})$$

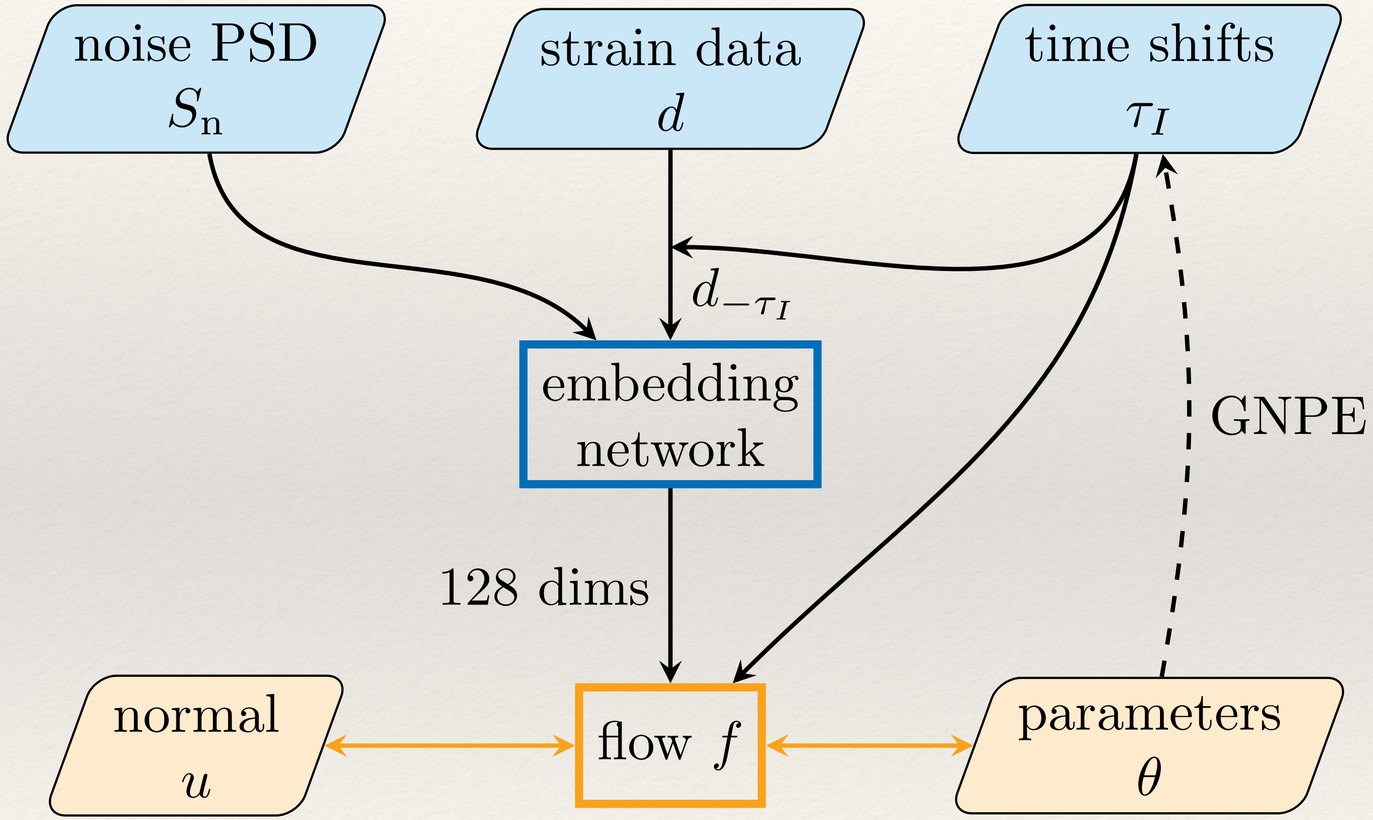
- Rather than enforce a covariant q_ϕ instead standardize data to have $t_c = 0$.
- However to know how much to shift the data, $t_c \in \theta$ must first be inferred.

Solution: Iterative transformations



Group-Equivariant NPE

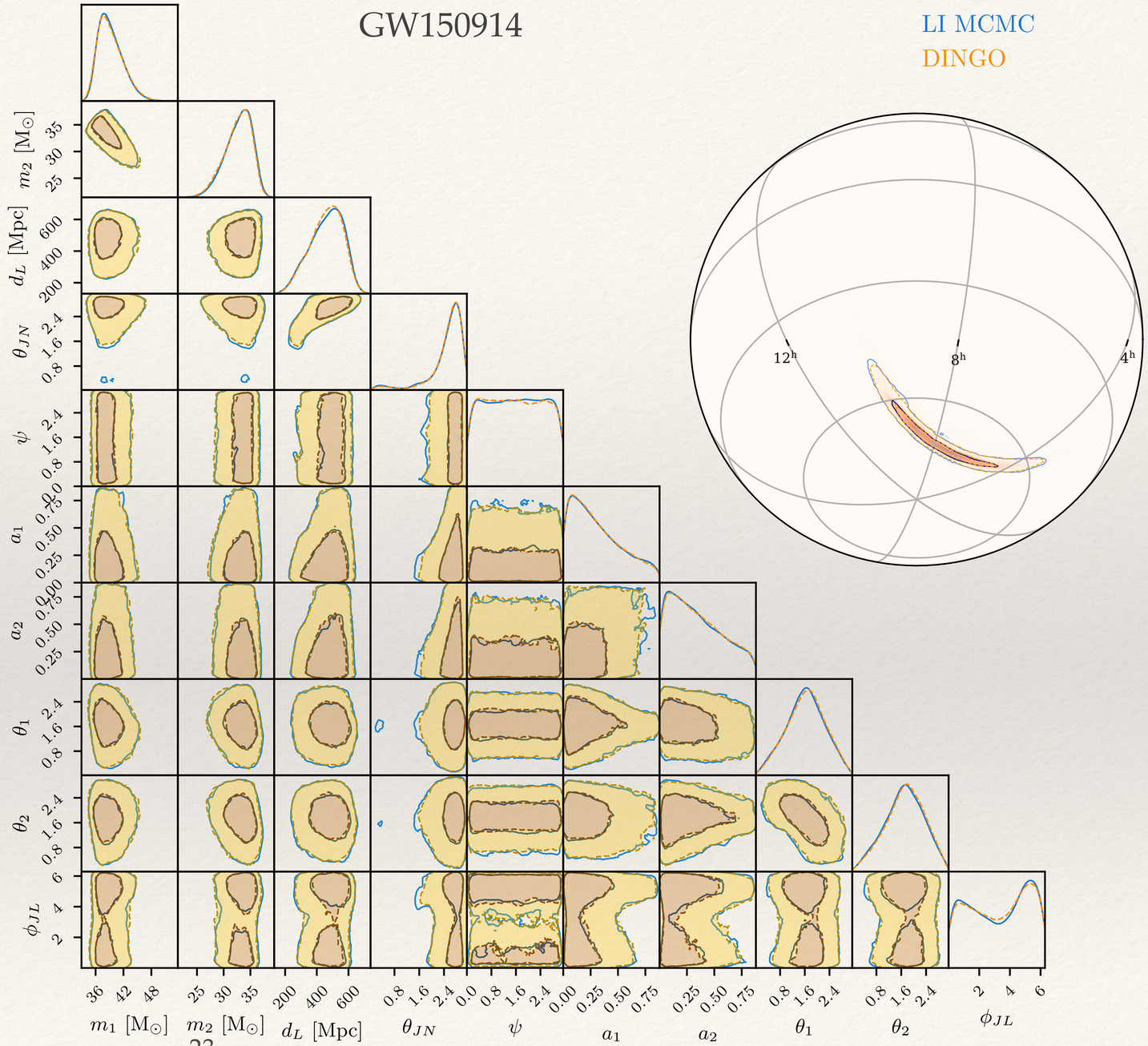
Dax, SRG+ (ICLR, 2022)



Dax, SRG+ (PRL 2021)

Extremely good agreement with standard techniques

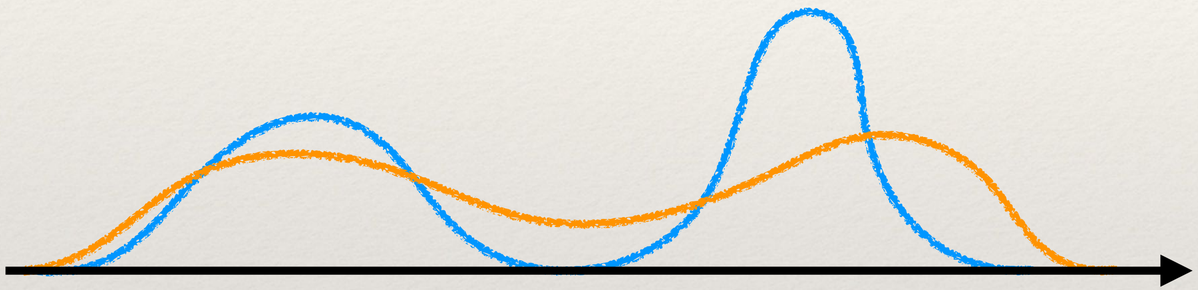
- $\sim 10^7$ training examples
  $\sim 10^8$ network parameters
- Training ~ few days
  Inference ~ minute



Verify results

Dax, SRG+ (PRL 2023)

- Since we have likelihood **and** the NPE density, we can use **importance sampling** to compare.



target (prior x likelihood)

$$w_i \propto \frac{p(\theta_i)p(d\,|\,\theta_i)}{q_\phi(\theta_i\,|\,d)}$$

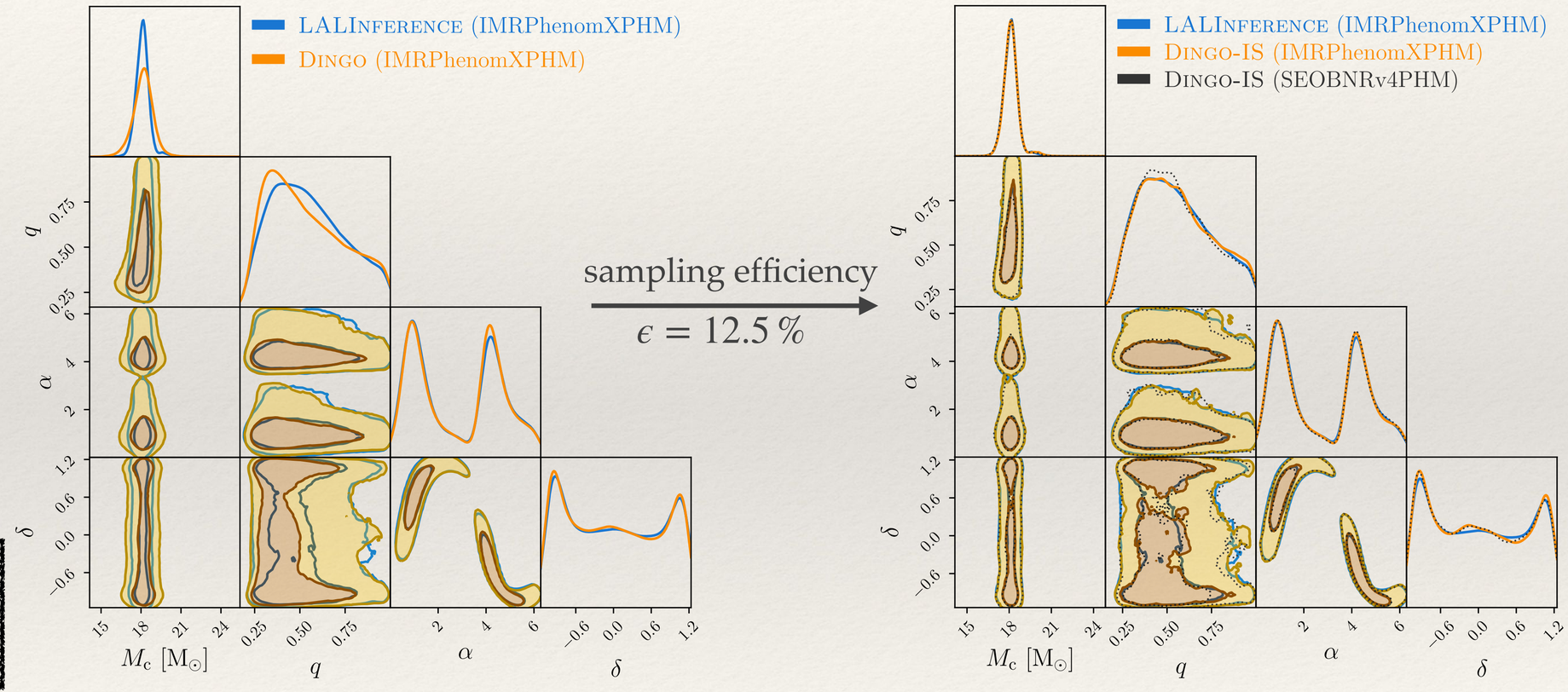
proposal (NPE)

- **Effective number of samples** $n_{\text{eff}} = \dfrac{\left(\sum_i w_i\right)^2}{\sum_i w_i^2}$ as measure of performance.
- **Evidence** $p(d) \approx \dfrac{1}{n}\sum_{i=1}^{n} w_i$

Neural importance sampling

Dax, SRG+ (2210.05686)

GW151012



Log evidence

Dingo-IS: -16412.88 ± 0.01

Bilby: -16412.73 ± 0.12

Flagging failure cases

- Sample efficiency ϵ serves to validate results.
- **Out-of-distribution data:**
  - Inconsistent with noise or signal model.
  - **Identifies** events with known issues with data quality or modeling.

Event	$\log p(d)$	ϵ	Event	$\log p(d)$	ϵ	Event	$\log p(d)$	ϵ
GW190408	-16178.332 ± 0.012	6.9%	GW190727	-15992.017 ± 0.009	10.3%	GW191230	-15913.798 ± 0.009	12.2%
_181802	-16178.172 ± 0.010	9.3%	_060333	-15992.428 ± 0.005	30.8%	_180458	-15913.918 ± 0.010	8.8%
GW190413	-15571.413 ± 0.006	22.5%	GW190731	-16376.777 ± 0.005	32.6%	GW200128	-16305.128 ± 0.013	6.1%
_052954	-15571.391 ± 0.005	26.3%	_140936	-16376.763 ± 0.005	31.0%	_022011	-16304.510 ± 0.007	18.3%
GW190413	-16399.331 ± 0.009	12.4%	GW190803	-16132.409 ± 0.006	21.4%	‡GW200129	-16226.851 ± 0.109	0.1%
_134308	-16399.139 ± 0.014	4.7%	_022701	-16132.408 ± 0.005	27.8%	_065458	-16231.203 ± 0.051	0.4%
GW190421	-15983.248 ± 0.008	15.3%	GW190805	-16073.261 ± 0.006	20.0%	GW200208	-16136.381 ± 0.007	16.6%
_213856	-15983.131 ± 0.010	9.4%	_211137	-16073.656 ± 0.007	16.6%	_130117	-16136.531 ± 0.009	11.2%
GW190503	-16582.865 ± 0.022	2.0%	GW190828	-16137.220 ± 0.009	12.2%	GW200208	-16775.200 ± 0.011	7.4%
_185404	-16583.352 ± 0.027	1.4%	_063405	-16136.799 ± 0.010	9.1%	_222617	-16774.582 ± 0.021	2.2%
GW190513	-15946.462 ± 0.043	0.6%	GW190909	-16061.634 ± 0.011	7.4%	GW200209	-16383.847 ± 0.009	12.5%
_205428	-15946.581 ± 0.017	3.4%	_114149	-16061.275 ± 0.016	3.8%	_085452	-16384.157 ± 0.025	1.6%
GW190514	-16556.466 ± 0.009	11.6%	GW190915	-16083.960 ± 0.015	20.8%	GW200216	-16215.703 ± 0.017	3.4%
_065416	-16556.314 ± 0.017	3.5%	_235702	-16083.937 ± 0.027	4.8%	_220804	-16215.540 ± 0.018	3.1%
GW190517	-16271.048 ± 0.027	1.3%	GW190926	-16015.813 ± 0.019	2.8%	GW200219	-16133.457 ± 0.011	9.6%
_055101	-16272.428 ± 0.034	0.9%	_050336	-16015.861 ± 0.009	12.1%	_094415	-16133.157 ± 0.017	4.0%
GW190519	-15991.171 ± 0.008	15.2%	GW190929	-16146.666 ± 0.018	3.2%	GW200220	-16303.782 ± 0.007	17.3%
_153544	-15991.287 ± 0.068	0.2%	_012149	-16146.591 ± 0.021	2.4%	_061928	-16303.087 ± 0.026	1.5%
GW190521	-16008.876 ± 0.008	13.4%	GW191109	-17925.064 ± 0.025	1.7%	GW200220	-16136.600 ± 0.008	13.2%
_074359	-16008.037 ± 0.015	4.2%	_010717	-17922.762 ± 0.041	0.6%	_124850	-16136.519 ± 0.037	0.7%
GW190527	-16119.012 ± 0.008	13.8%	GW191127	-16759.328 ± 0.019	2.7%	GW200224	-16138.613 ± 0.006	22.5%
_092055	-16118.781 ± 0.013	6.1%	_050227	-16758.102 ± 0.029	1.2%	_222234	-16139.101 ± 0.006	21.4%
GW190602	-16036.993 ± 0.006	25.0%	‡GW191204	-15984.455 ± 0.015	4.2%	‡GW200308	-16173.938 ± 0.013	6.0%
_175927	-16037.529 ± 0.006	23.5%	_110529	-15983.618 ± 0.063	0.3%	_173609	-16173.692 ± 0.025	1.7%
GW190701	-16521.381 ± 0.040	0.6%	GW191215	-16001.286 ± 0.013	5.8%	GW200311	-16117.505 ± 0.011	7.4%
_203306	-16521.609 ± 0.010	10.1%	_223052	-16000.846 ± 0.052	0.4%	_115853	-16117.583 ± 0.009	11.9%
GW190719	-15850.492 ± 0.008	13.4%	GW191222	-15871.521 ± 0.007	16.5%	‡GW200322	-16313.568 ± 0.307	0.0%
_215514	-15850.339 ± 0.011	8.0%	_033537	-15871.450 ± 0.005	25.8%	_091133	-16313.110 ± 0.105	0.1%

Table II. 42 BBH events from GWTC-3 analyzed with DINGO-IS. We report the log evidence $\log p(d)$ and the sample efficiency ϵ for the two waveform models IMRPhenomXPHM (upper rows) and SEOBNRv4PHM (lower rows). Highlighting colors indicate the sample efficiency (green: high; yellow: medium; orange/red: low); DINGO-IS results can be trusted for medium and high ϵ (see Supplemental Material). Events in gray suffer from data quality issues [1, 21]. ‡See remarks on these events in text.

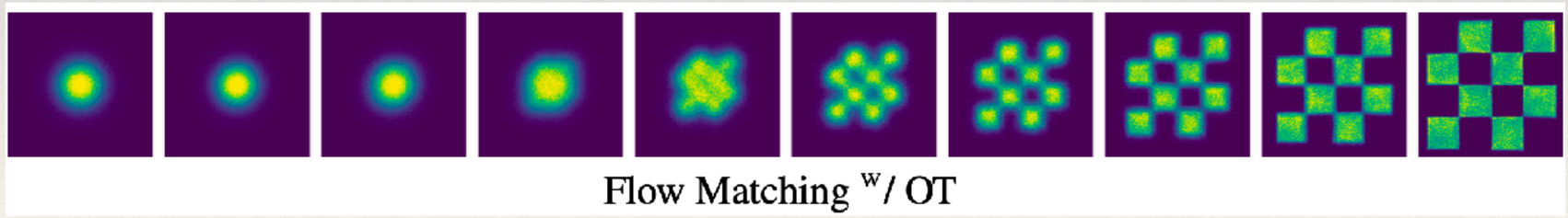
What's next?

- **In principle, this is arbitrarily flexible!** Any data, any representation, any parameters…

  … But this requires bigger networks.

- We find that (discrete) normalizing flows do not scale well beyond $\sim 10^8$ trainable parameters and ~ 15 inference parameters.

- Modern generative architectures commonly used for image generation are simpler and have better scaling. **Flow matching** [Lipman et al (ICLR 2023)] uses a continuous normalizing flow.



$t = 0 \longrightarrow t \qquad t = 1$

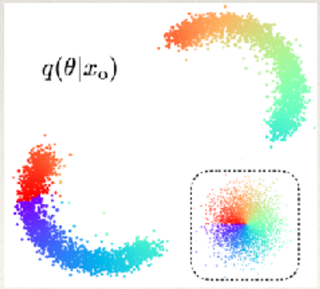
Flow matching posterior estimation

- FMPE uses flow matching to estimate the posterior

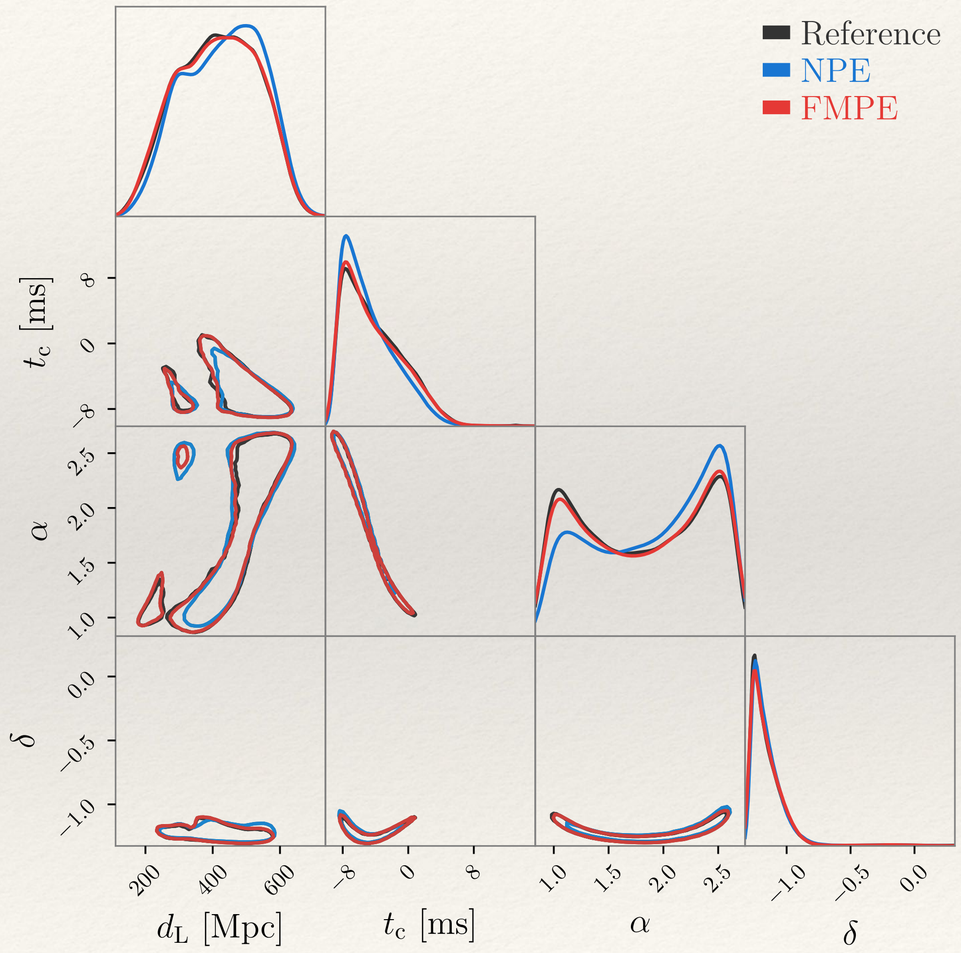
$$\frac{d\theta_t}{dt} = v_{t,d}(\theta_t)$$

Network learns vector field

- Outperforms NPE with discrete flows
  - Faster training, better scaling to large networks
  - Intermediate $0 < t < 1$ are more interpretable



Dax, Wildberger, Buchholz, SRG+ (NeurIPS 2023)



Population inference

Leyde, Green, Toubiana, Gair (2023)

Given a collection of observations $\{d^{(i)}\}_{i=1}^{N}$ determine general properties of the population, e.g., mass distribution.

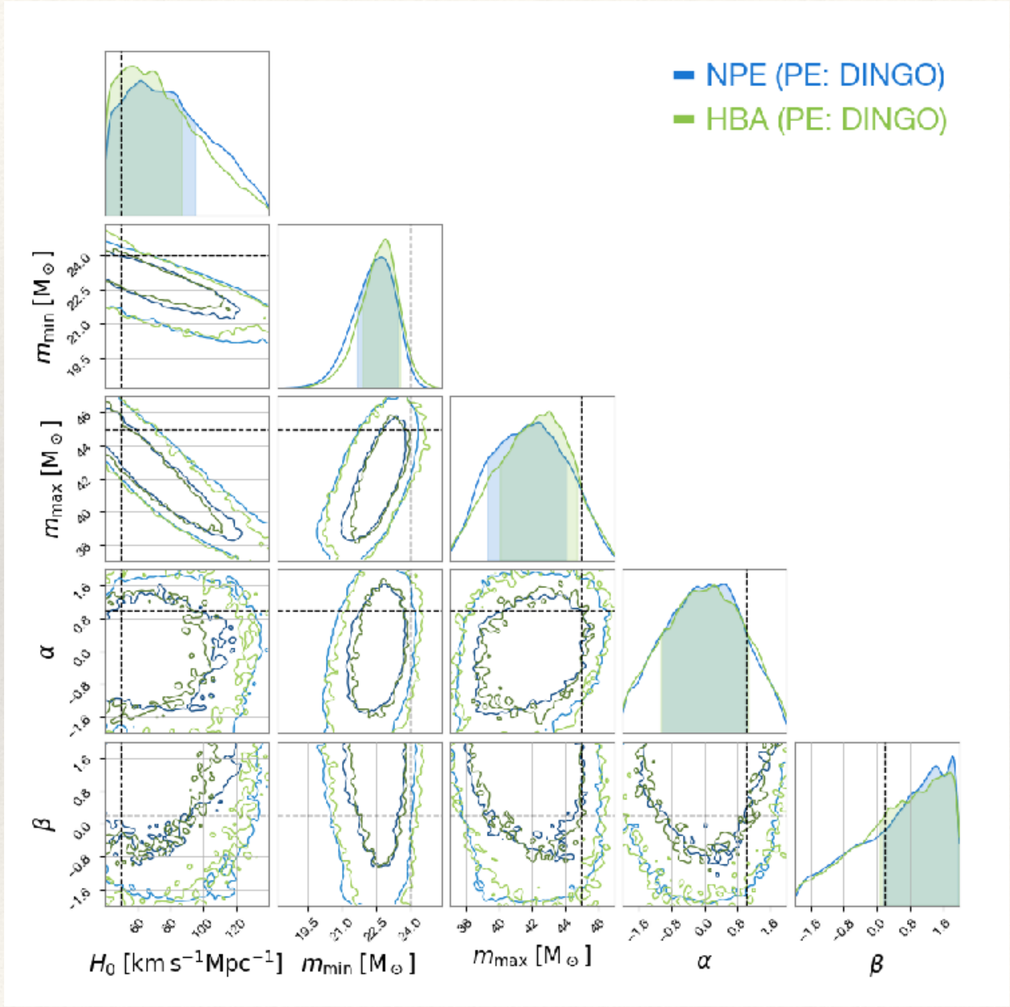
*hyper*parameters

- Population likelihood $p_{\text{pop}}(\theta \mid \Lambda)$
- **Hierarchical Bayesian inference** gives population posterior

$$p(\Lambda \mid \{d^{(i)}\}) = \frac{p(\Lambda)}{p(\{d^{(i)}\})} \prod_{j=1}^{N} \frac{\int p(d_j \mid \theta_j)\, p_{\text{pop}}(\theta_j \mid \Lambda)\, \mathrm{d}\theta_j}{\int p_{\text{det}}(\theta_j)\, p_{\text{pop}}(\theta_j \mid \Lambda)\, \mathrm{d}\theta_j}$$

Selection effects

Learn directly with NPE!



Conclusions

- **Accurate inference for binary black holes in seconds to minutes.**
  - Enable **rapid alerts** and **huge numbers of events / analyses.**
  - Classical techniques like **importance sampling** can be used to validate results.
- **Ready to be used:** Code available @ https://github.com/dingo-gw/dingo
- **Outlook**
  - Exciting prospects such as noise-model-free inference, populations and cosmology, and 3G/LISA inference.
  - New architectures likely to deliver ever-improving performance.

Thank You!