DeepHMC: a deep learning Hamiltonian Monte Carlo sampler for faster Bayesian inference of GWs

Meeting of the National Research Group on Gravitational Waves

- Data analysis method session -

Marc Arène, Ed Porter

30/03/2021

Based on document: LIGO-G2100538

Outline

I. Building a competitive Hamiltonian Monte Carlo (HMC)

II. DeepHMC: a DNN boosted HMC

III. Comparison with LALInferenceMCMC

Current Parameter Estimation in the LVK

- Parallel tempered MCMC and Nested sampling: **random walk samplers.**

- Current statuts:

	BBH	BNS	NSBH
Signal duration	$\sim 1 - 8\,s$	$\sim 64 - 256\,s$	$\sim 2 - 128\,s$
PE duration	Few days	Few weeks - months	Few weeks - months

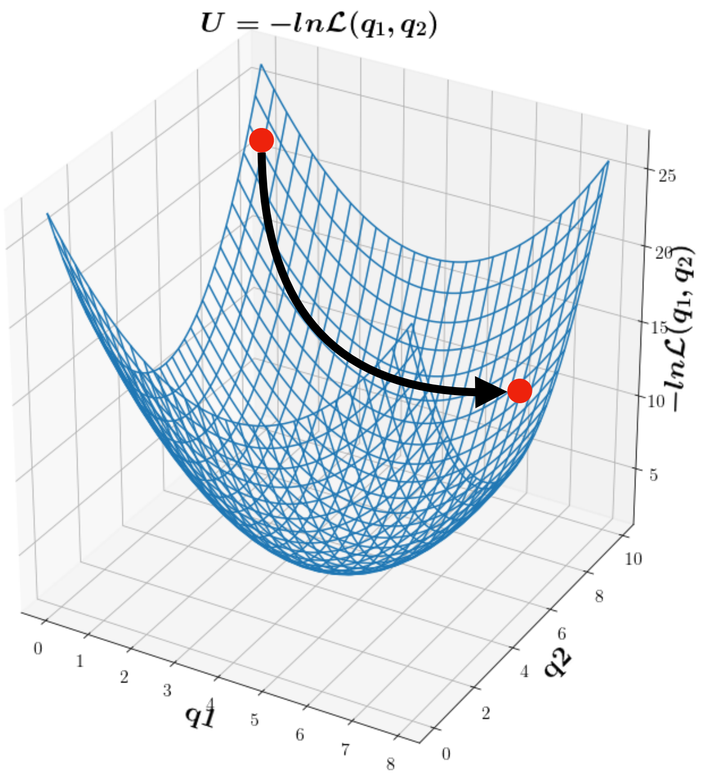
- Likelihood is evaluated ~ 10^8 times (for MCMC) to get an effective sample size of 10^4 statistically independent samples (SIS).

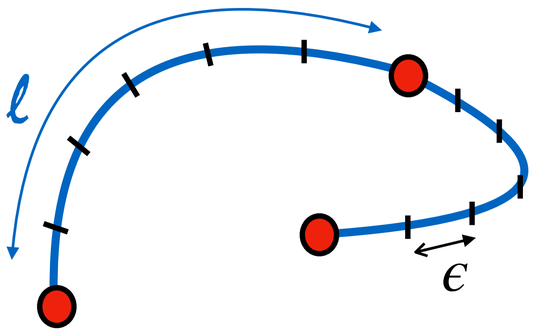
- As interferometers are being improved, this implies in the future…
  - Longer signals to analyze.
  - More detections per year: 10^{+52}_{-10} BNS . y^{-1} for O4.

The LIGO Scientific Collaboration, et al. Prospects for Observing and Localizing Gravitational-Wave Transients with Advanced LIGO, Advanced Virgo and KAGRA. arXiv:1304.0670 [astro-ph, physics:gr-qc], January 2020.

Current samplers vs HMC

MCMC / Nested sampling	Hamiltonian Monte Carlo (HMC)
• Easy to code up. • Convergence is assured.	• Non-random-walk sampler. • Simulates Hamiltonian trajectories using $\partial_\mu \ln \mathscr{L}$ to move. • D times more efficient than random-walk MCMC.
• Random-walk samplers. • Convergence time undefined.	• Hard to set-up, fine tune internal parameters. • Numerical gradients, $(\partial_\mu \ln \mathscr{L})_{num}$, are **very expensive when no closed form solution exists**.





I

Building a competitive HMC

I. Building a competitive HMC

- At each step of the trajectory, replace numerical gradient by a cubic fit approximation (Carré and Porter, 2013, arxiv.org/abs/1311.7539)

$$(\partial_\mu \ln \mathscr{L})_{cubic}(q^\mu) = a_0 + \sum_{i=1}^{D} a_i q^i + \sum_{j=1}^{D} \sum_{k=j}^{D} a_{jk} q^j q^k + \sum_{l=1}^{D} \sum_{v=l}^{D} \sum_{w=v}^{D} a_{lvw} q^l q^v q^w$$

DxD symmetric matrix **DxDxD symmetric tensor**

- 455 coefficients per dimension if D = 12.

- Cubic fit fails for $\{\cos\theta_{JN}, \psi, \ln D_L\}$ when $\cos\theta_{JN}$ is bi-modal so we replace it with Ordered Look-Up Tables (OLUTs) to produce a local fit at each step (Bouffanais & Porter, 2018, arxiv.org/abs/1810.07443).

I. Building a competitive HMC

- GW170817
- $f_{low} = 30$Hz thus 64 sec of data, $f_{samp} = 4096$Hz
- Aligned-spin with `IMRPhenomD_NRTidal`
- Marginalization over phase at coalescence
- $D = 12$
- GWTC-1 PSDs used
- No marginalization over calibration uncertainties
- No “burn-in” of the chain

Parameter	Shape of prior	Limits
θ_{JN}	Sinusoidal	0–π
ψ	Uniform	0–π
D_L	Quadratic	10–100
m_1	Uniform	0.5–7.7
m_2	Uniform	0.5–7.7
δ	Cosinusoidal	$-\pi/2$–$\pi/2$
α	Uniform	0–2π
δt_c	Uniform	$\delta t_c^{det} \pm 0.1$
χ_1	zprior	-0.05–0.05
χ_2	zprior	-0.05–0.05
Λ_1	Uniform	0–5000
Λ_2	Uniform	0–5000

I. Building a competitive HMC

- Phase I
  - Run 1500 numerical gradient trajectories over 200 steps.
  - If trajectory is accepted, keep q^μ and $\left(\partial_\mu \ln \mathscr{L}\right)_{num}$ at each step.

$$Acc_{PhI} = 94\,\%$$

$$\left\{ q^\mu, \left(\partial_\mu \ln \mathscr{L}\right)_{num} \right\} : \; 281{,}200 \text{ samples}$$

$$\left(\partial_\mu \ln \mathscr{L}\right)_{num} : 642 \text{ ms} \quad \Rightarrow 54\,\text{h}$$

- Phase II
  - Derive the 455 coefficients of the cubic fit for each dimension, using a QR decomposition over the set of ~1500x200 $\left\{ q^\mu, \left(\partial_\mu \ln \mathscr{L}\right)_{num} \right\}$.
  - Create the OLUTs for $\{\cos\theta_{JN}, \psi, lnD_L\}$.

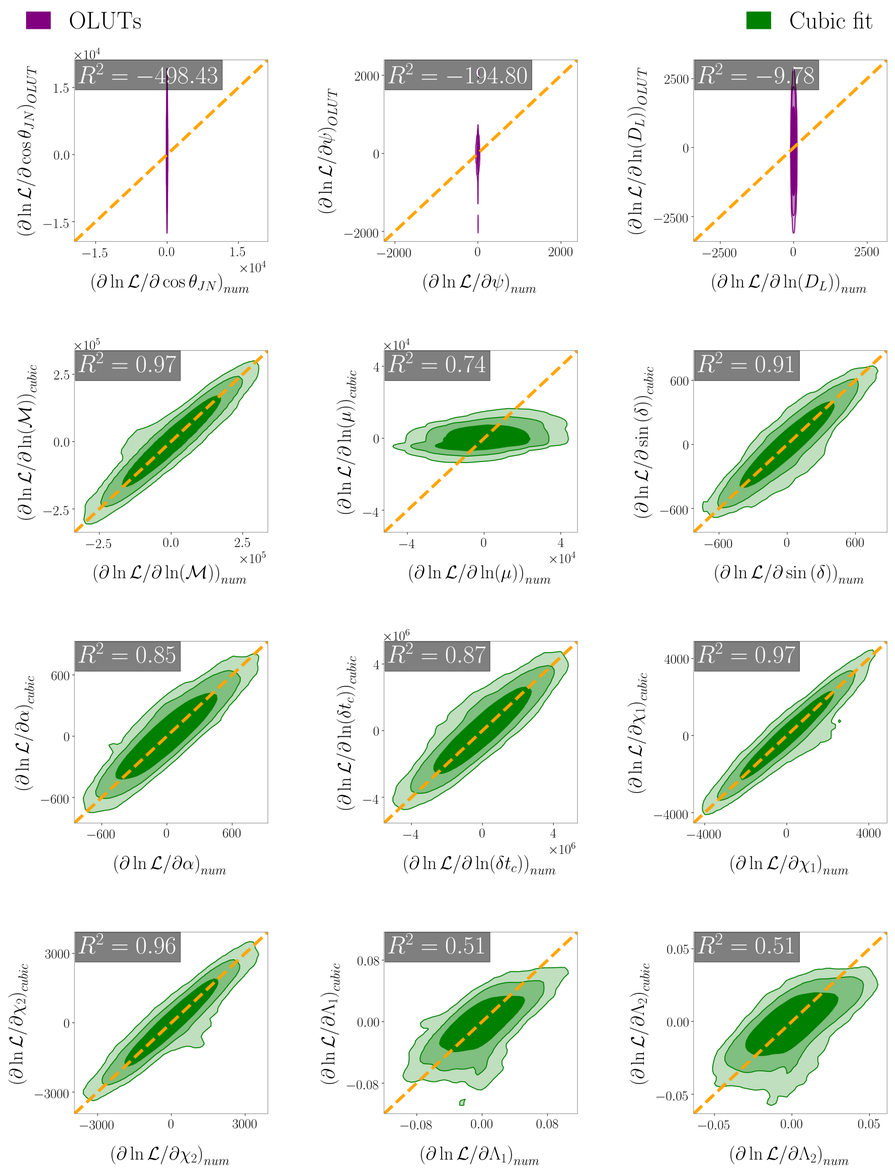
~ 7 min

- Phase III
  - Replace $\left(\partial_\mu \ln \mathscr{L}\right)_{num}$ with $\left\{ \left(\partial_\mu \ln \mathscr{L}\right)_{cubic}, \left(\partial_\mu \ln \mathscr{L}\right)_{OLUT} \right\}$.
  - At the end of each trajectory, compute the Metropolis-Hastings ratio numerically.

I. Building a competitive HMC

Predictive performances on a <u>validation set</u> of same size (281,200 data points) as that built during Phase I

(Contours encompass 90%, 99% and 99.9% of the data)



I. Building a competitive HMC

- $\left\{ \left(\partial_\mu \ln \mathscr{L}\right)_{cubic}, \left(\partial_\mu \ln \mathscr{L}\right)_{OLUT} \right\}$: 1.3 ms $\sim \times 500$ faster than $\left(\partial_\mu \ln \mathscr{L}\right)_{num}$

- $Acc_{PhIII} = 45\,\%$

- $N_{eff} = \left\lfloor \frac{N_{PhIII}}{ACL_{max}} \right\rfloor$

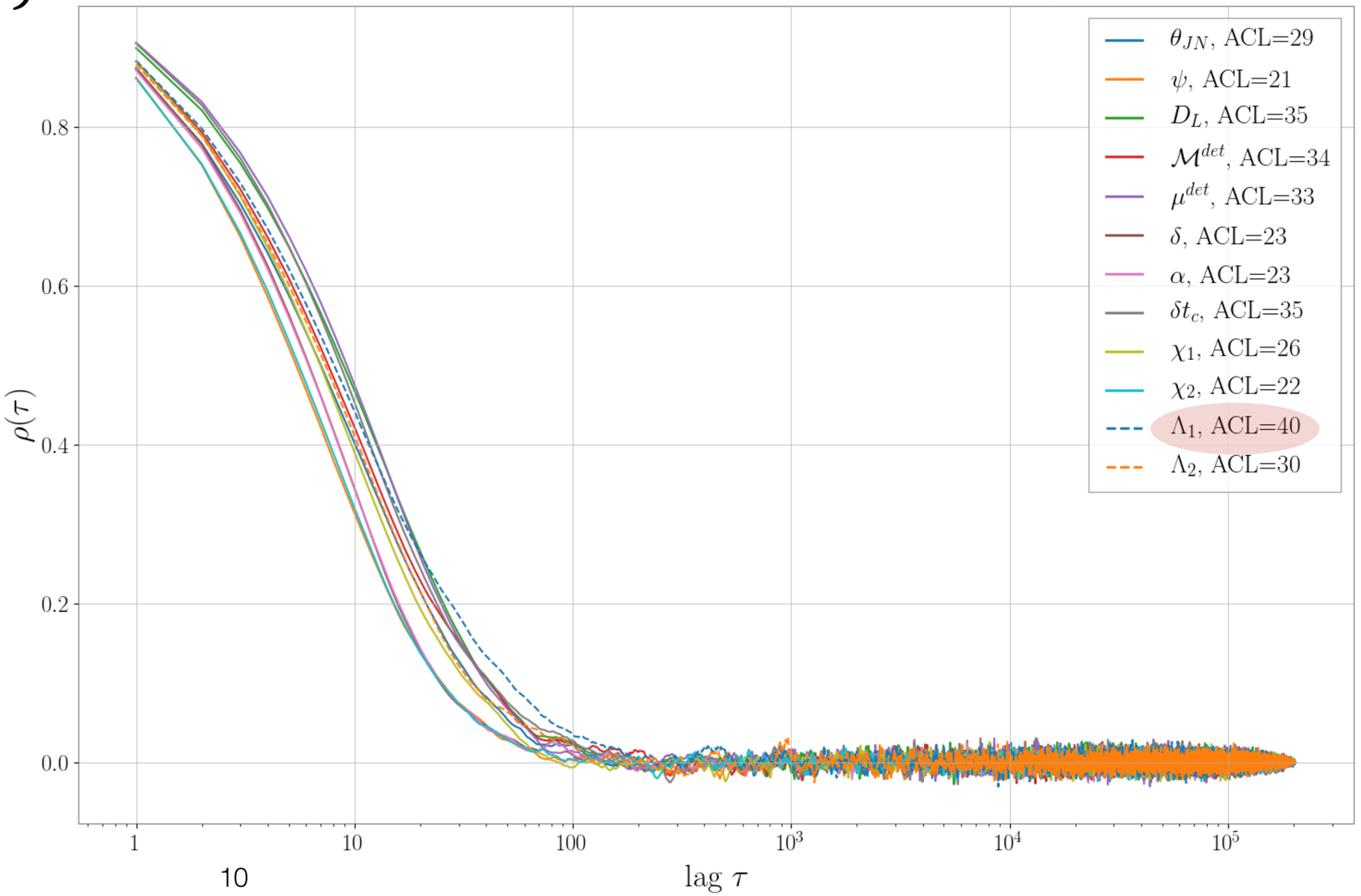
203,500

40

5,087 SIS

- Duration Phase III = **60.3 h**

Phase III



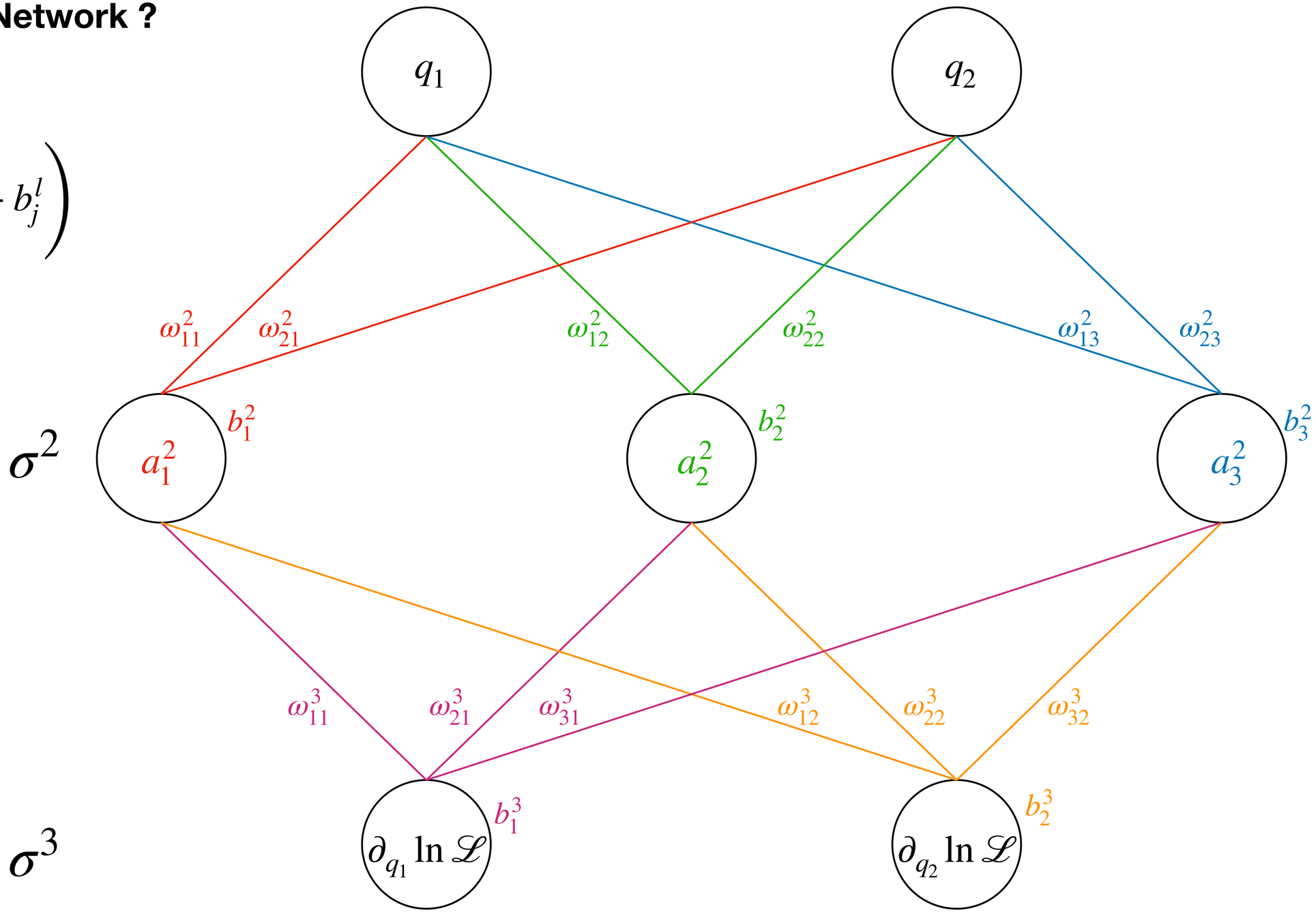
II

DeepHMC: a DNN boosted HMC

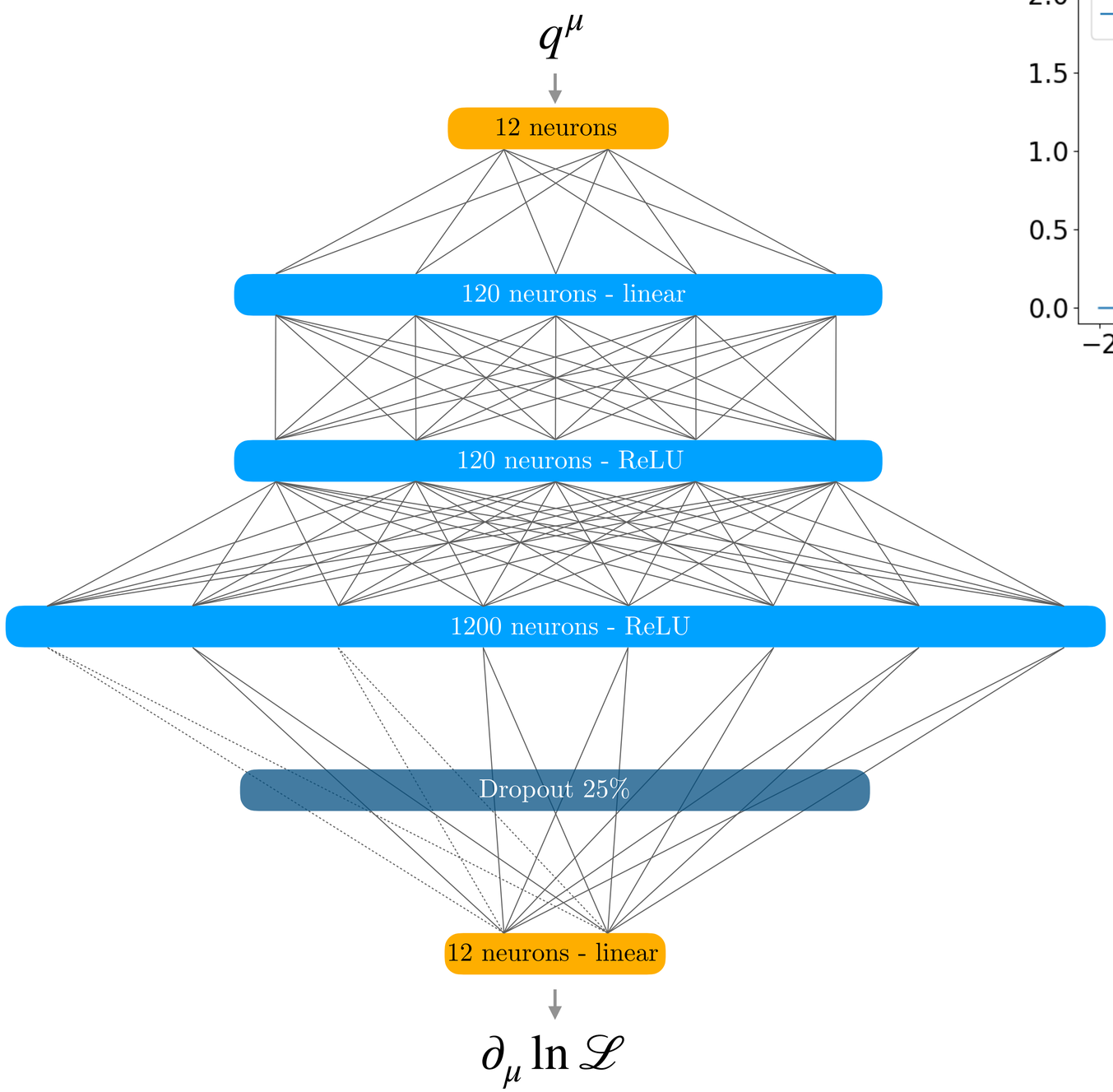
What is a Deep Neural Network ?

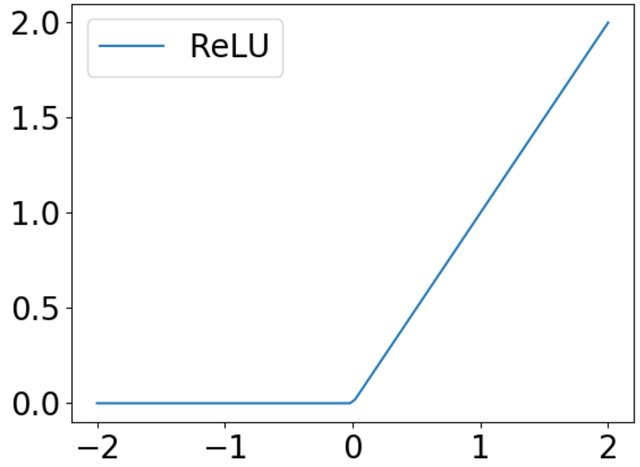
$$a_j^l = \sigma^l\left(\sum_{i=1}^{N(l-1)} w_{ij}^l\, a_i^{l-1} + b_j^l\right)$$

$$D = 2$$



$D = 12$

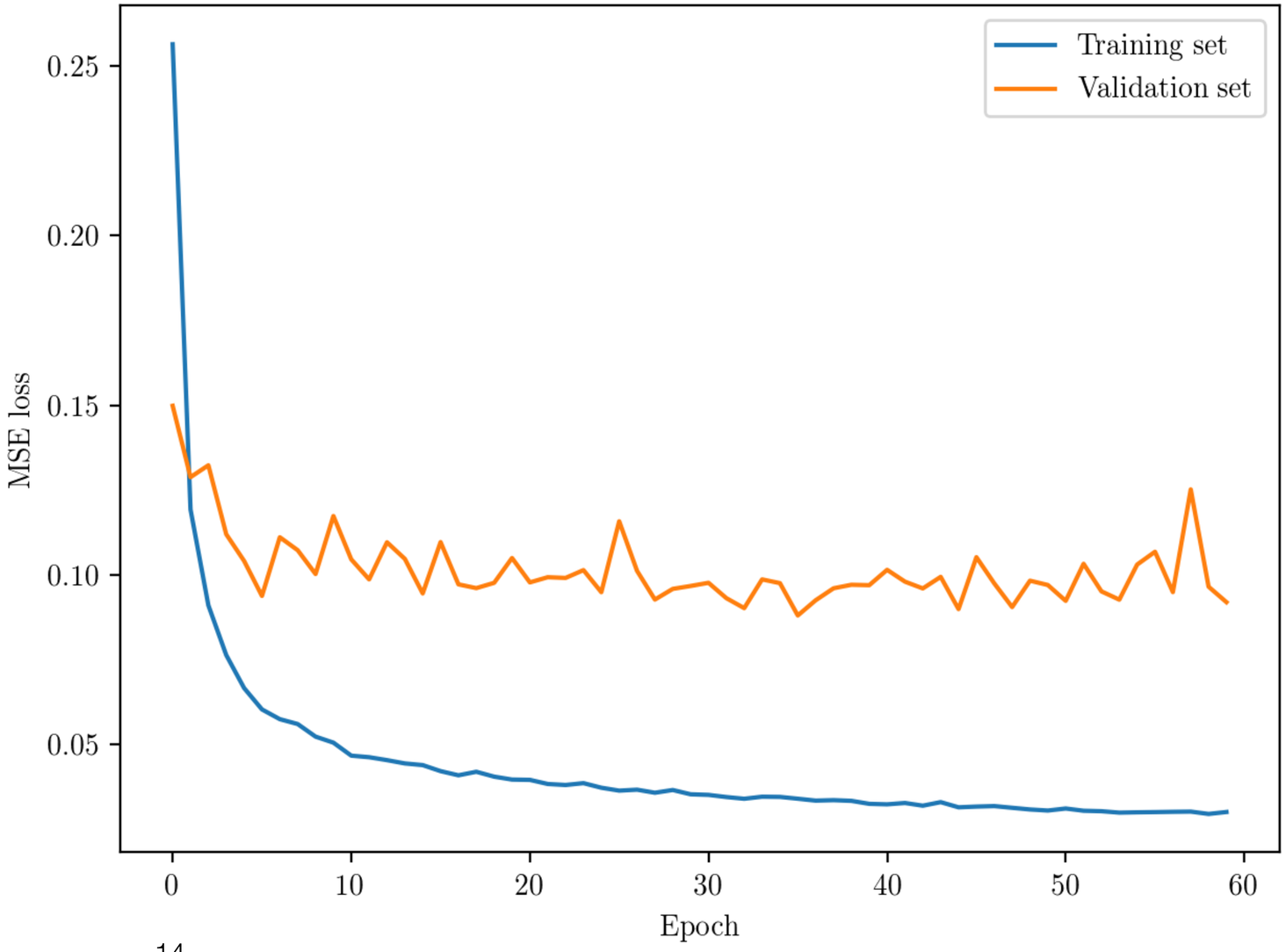

q^μ
12 neurons
120 neurons - linear
120 neurons - ReLU
1200 neurons - ReLU
Dropout 25%
12 neurons - linear
$\partial_\mu \ln \mathscr{L}$


ReLU
2.0
1.5
1.0
0.5
0.0
−2
−1
0
1
2

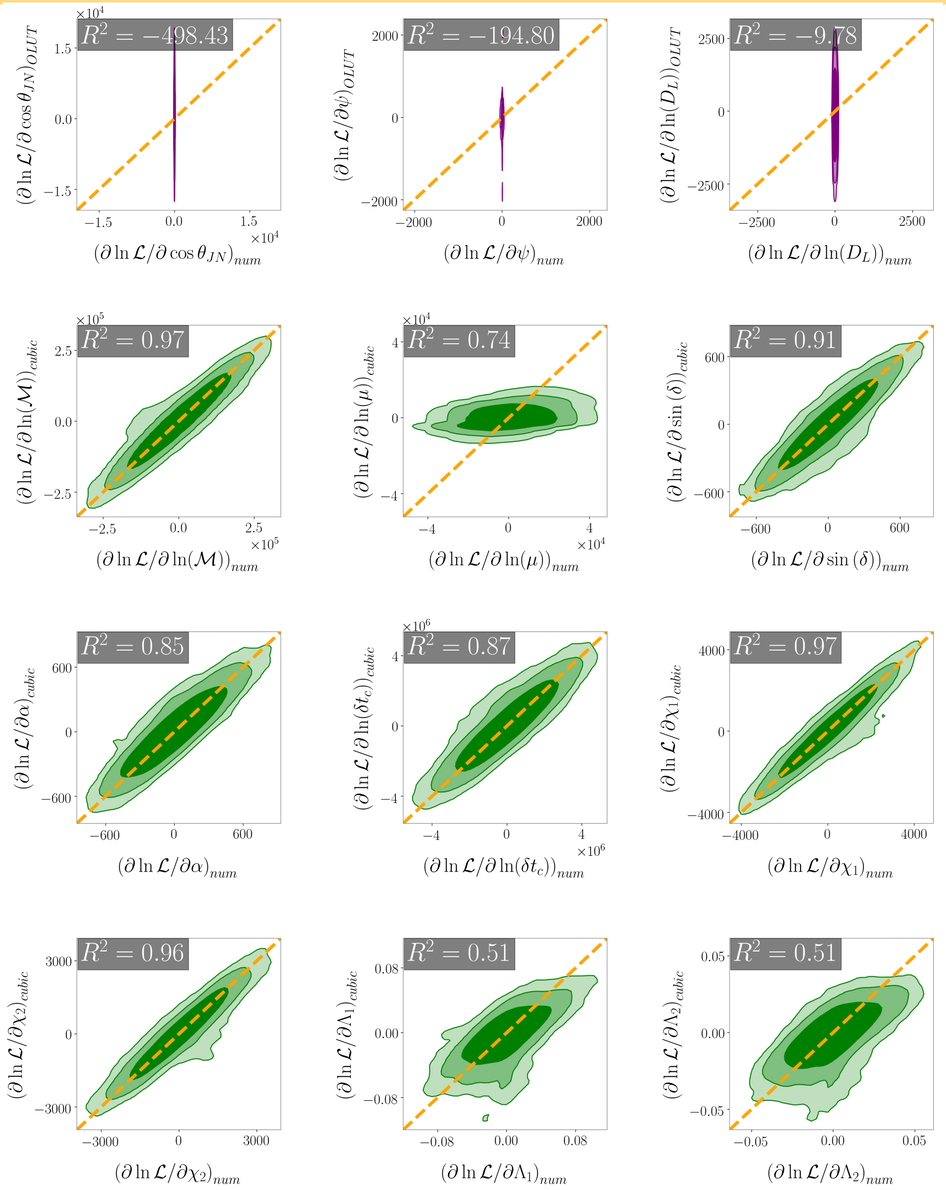
II. DeepHMC: a DNN boosted HMC

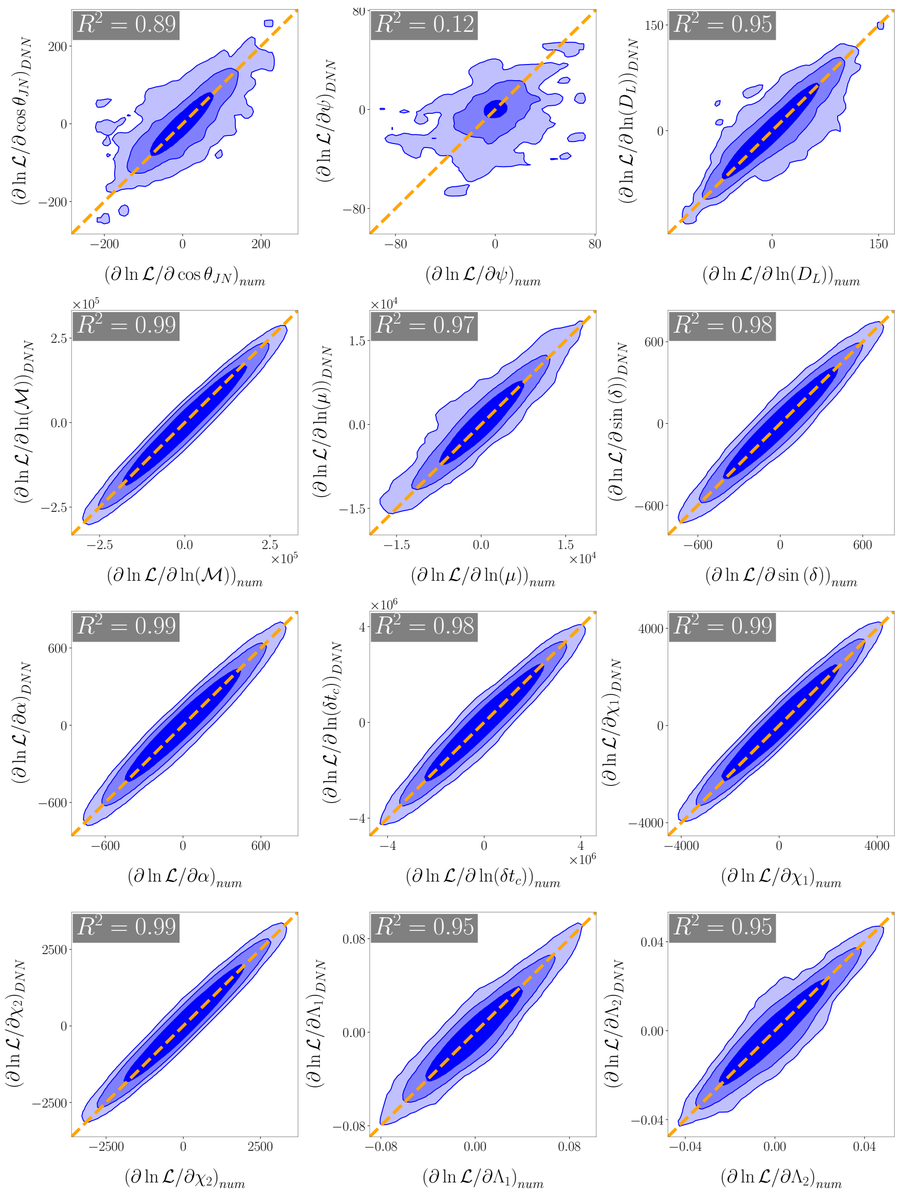
- Training set is split into batches of 128 samples randomly selected.
- Mean Squared Error (MSE) loss function for the gradient descent.
- 1 epoch is completed when the gradient descent is performed on all batches.
- Training duration = 12 min.

Phase II



II. DeepHMC: a DNN boosted HMC





II. DeepHMC: a DNN boosted HMC

Phase III

- $\left(\partial_\mu \ln \mathscr{L}\right)_{DNN}$: 0.5 ms

$\sim \times 2$ faster than $\left\{\left(\partial_\mu \ln \mathscr{L}\right)_{cubic}, \left(\partial_\mu \ln \mathscr{L}\right)_{OLUT}\right\}$

$\gtrsim \times 1000$ faster than $\left(\partial_\mu \ln \mathscr{L}\right)_{num}$

II. DeepHMC: a DNN boosted HMC

- $\left(\partial_\mu \ln \mathscr{L}\right)_{DNN}$: 0.5 ms

- $Acc_{PhIII} = 50\,\%$

- $N_{eff} = \left\lfloor \frac{N_{PhIII}}{ACL_{max}} \right\rfloor$

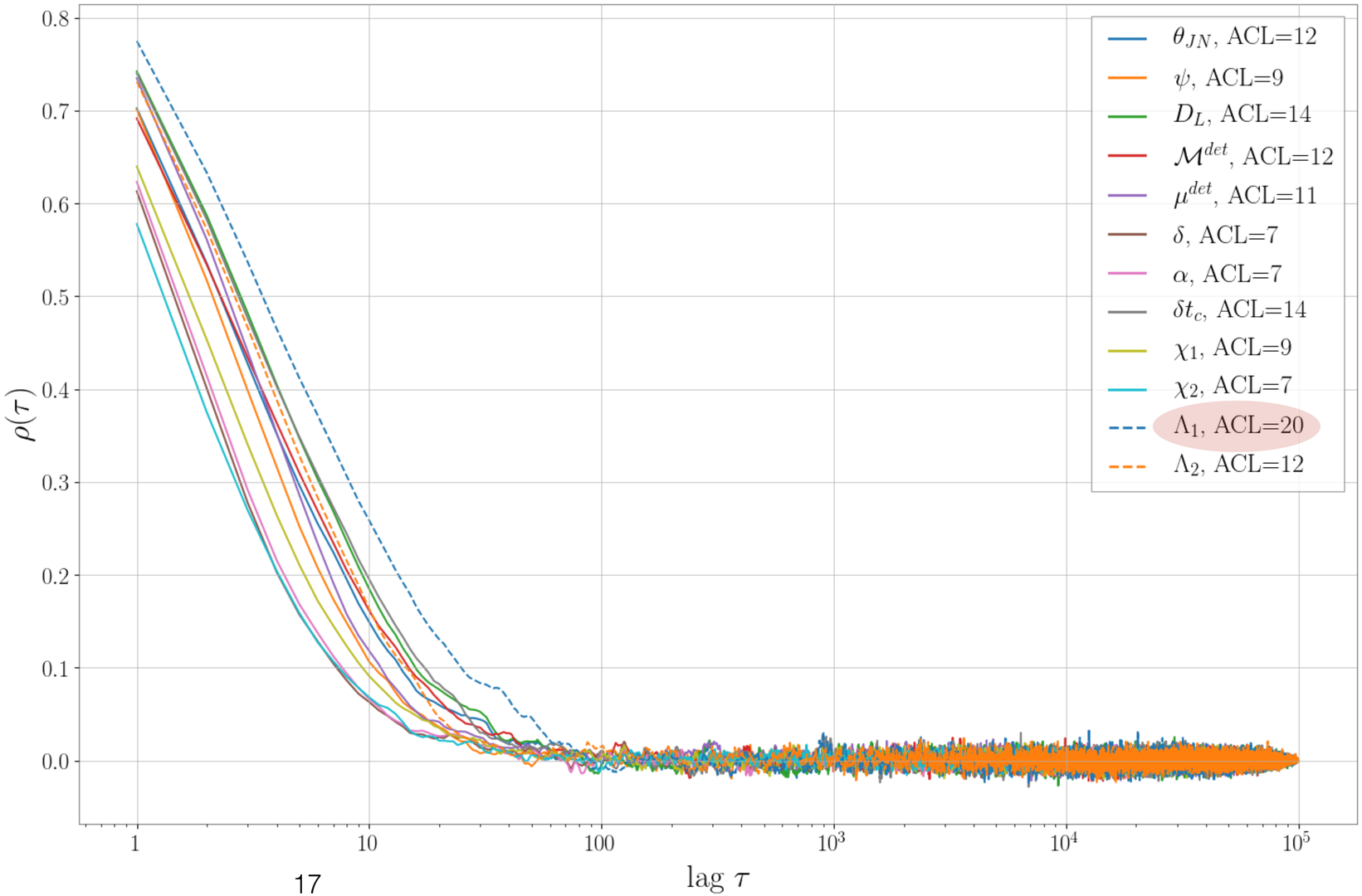
100,000

20

5,000 SIS

- Duration Phase III = **14.5 h**

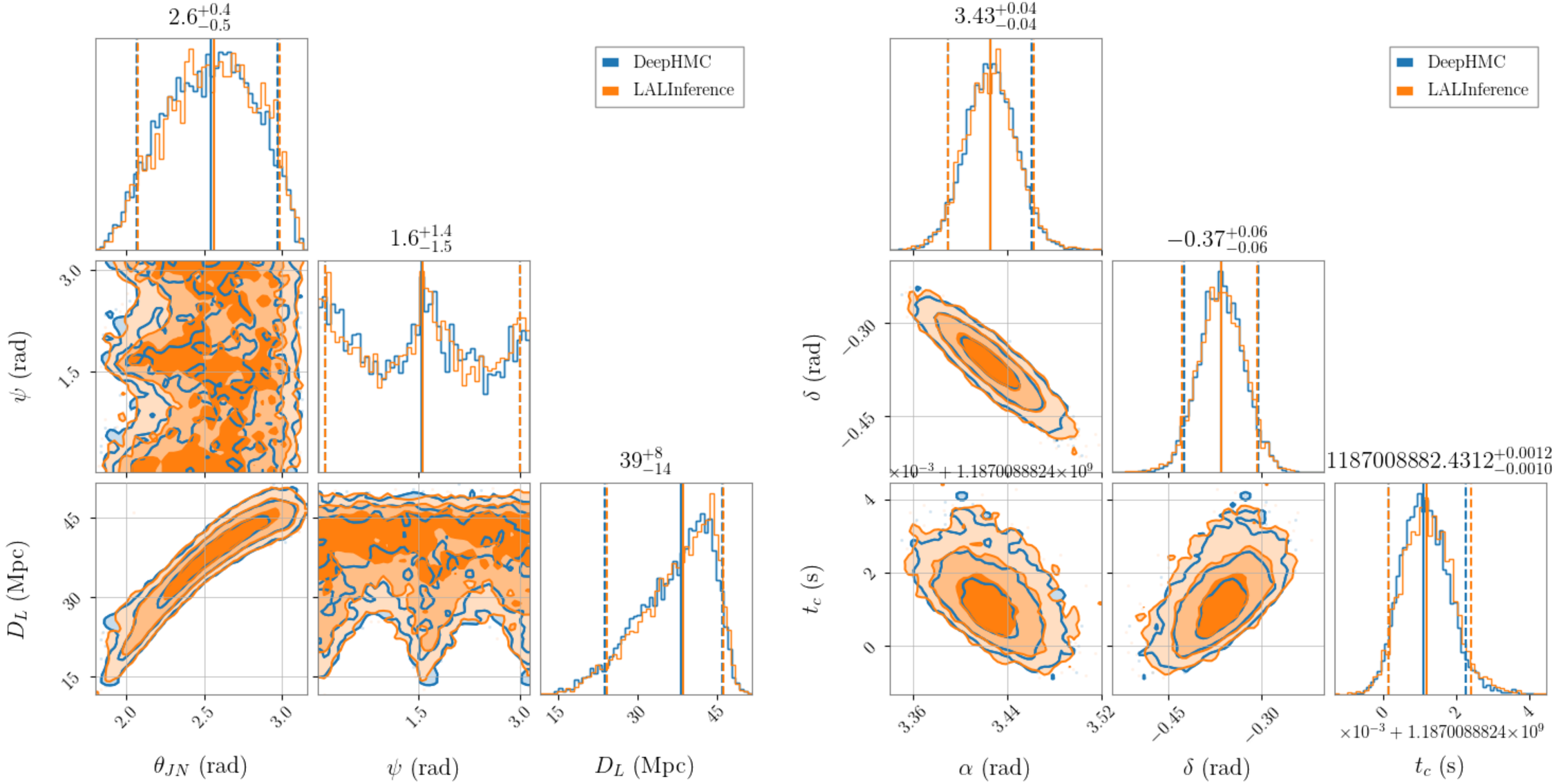
Phase III



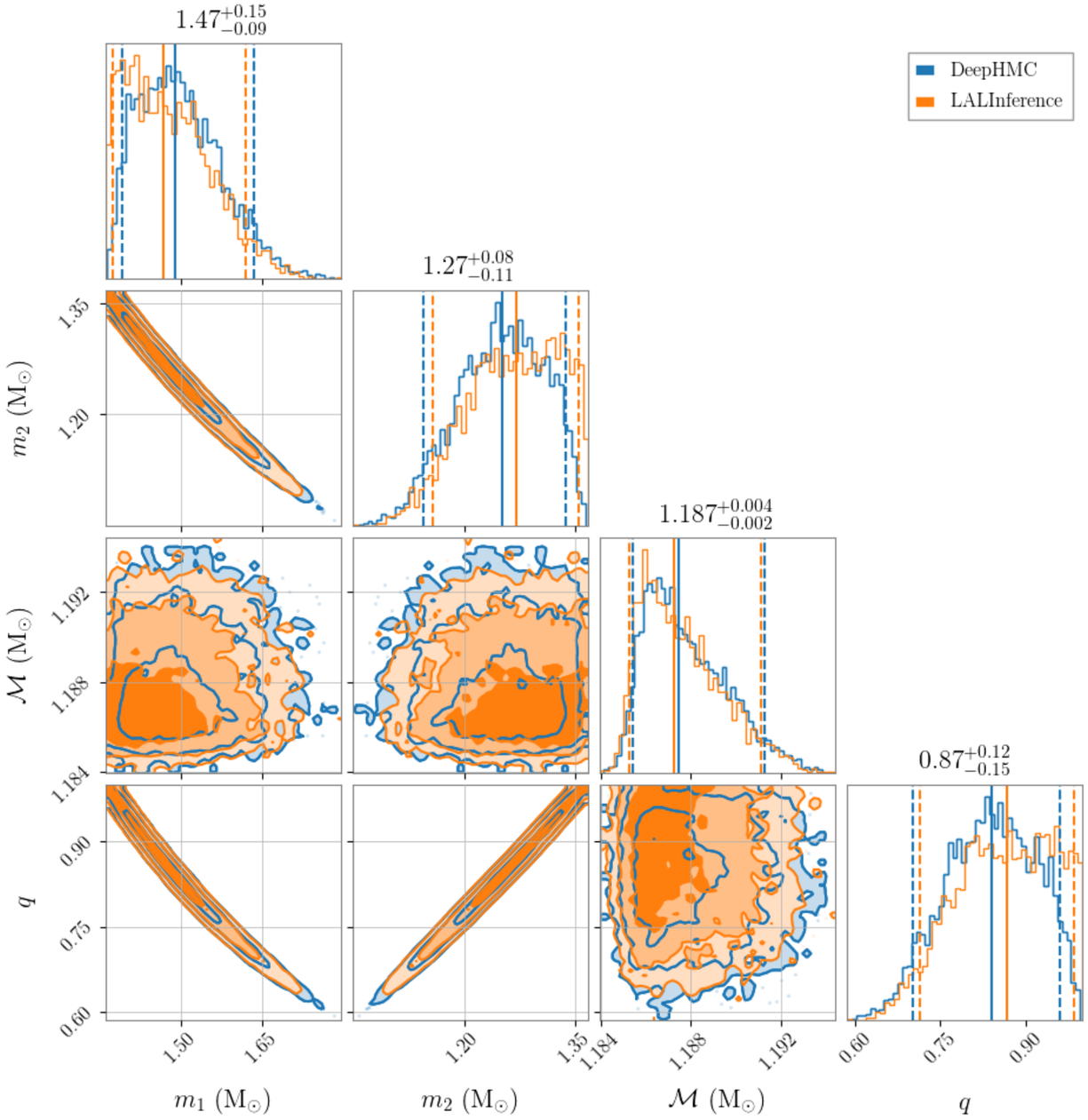
III

Comparison with LALInferenceMCMC

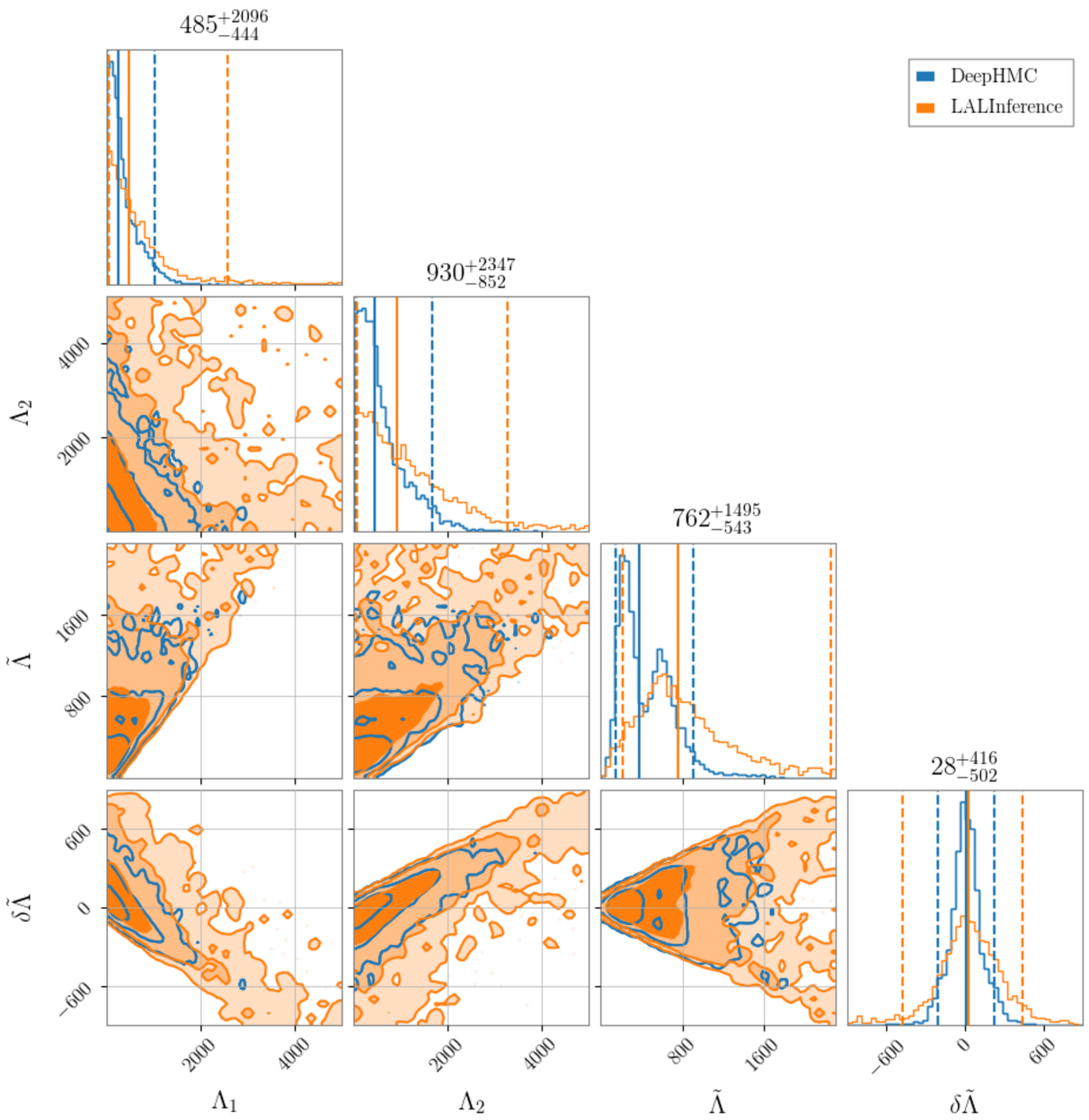
III. Comparison with LALInferenceMCMC

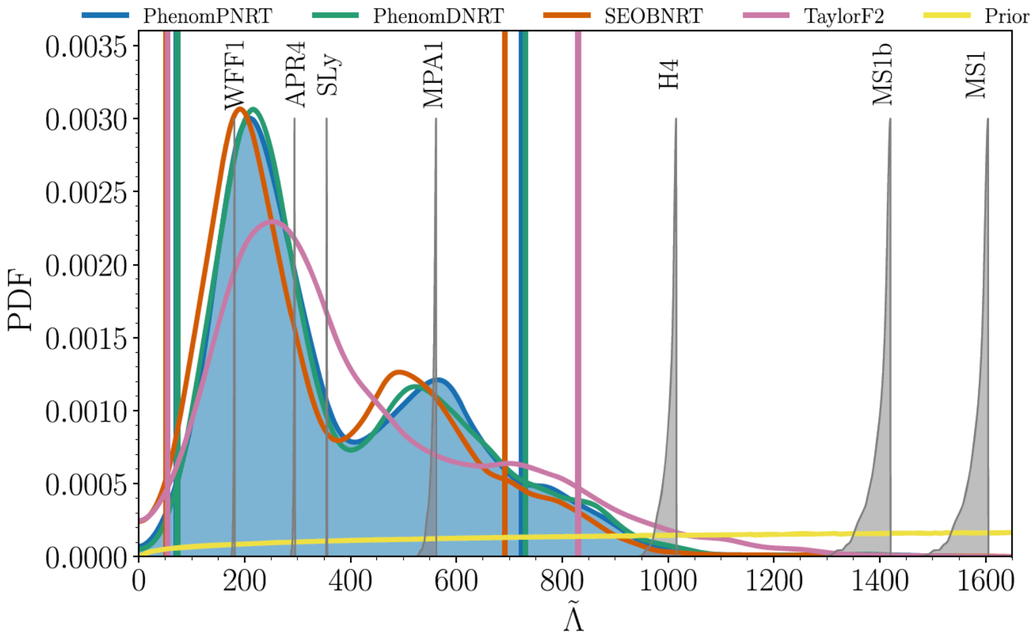


III. Comparison with LALInferenceMCMC



III. Comparison with LALInferenceMCMC



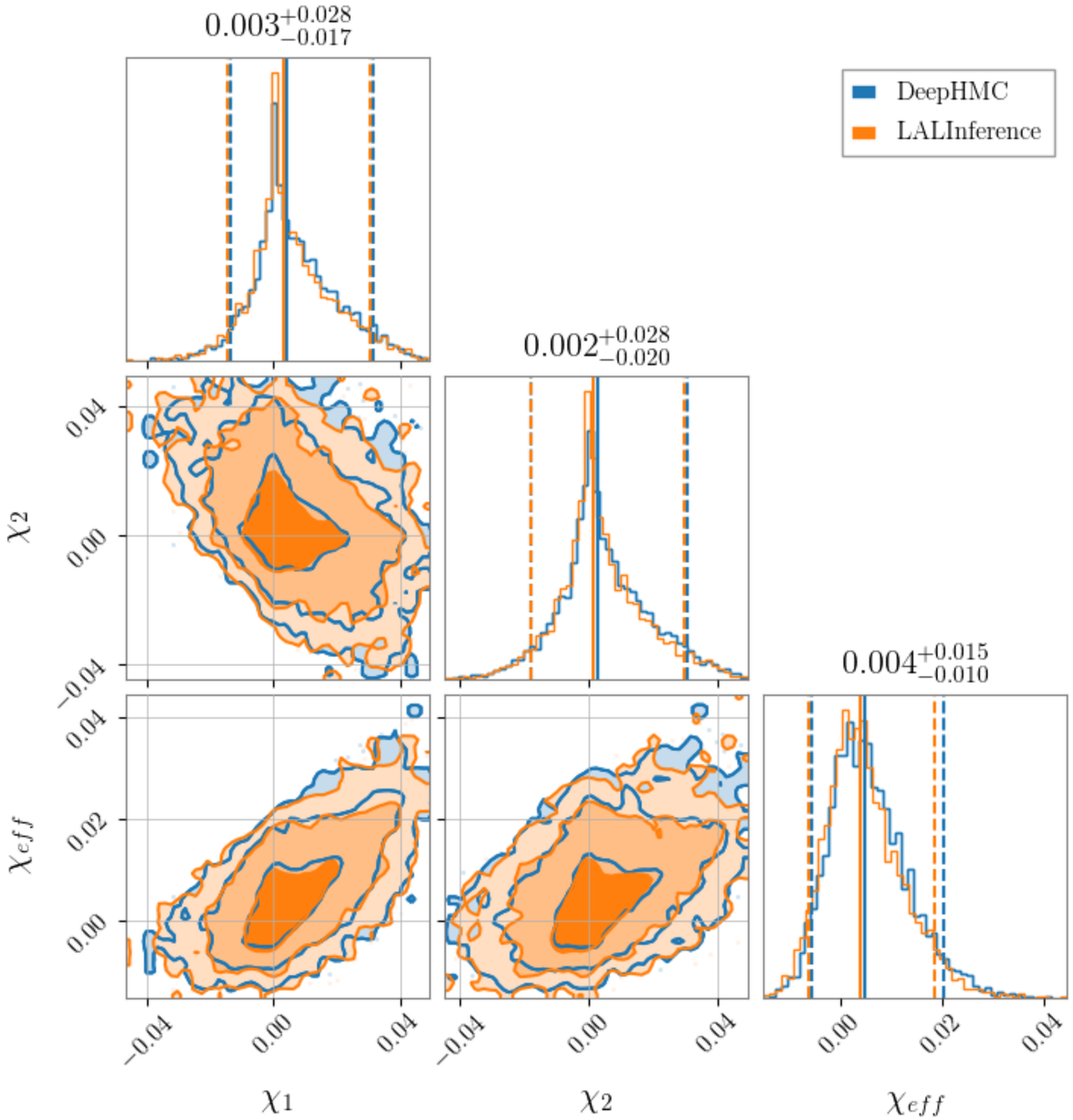


The LIGO Scientific Collaboration, et al. **Properties of the Binary Neutron Star Merger GW170817**. Physical Review X, 9(1):011001, January 2019.

$$\tilde{\Lambda} = \frac{8}{13}\left[\left(1+7\eta-31\eta^2\right)\left(\Lambda_1+\Lambda_2\right)+\sqrt{1-4\eta}\left(1+9\eta-11\eta^2\right)\left(\Lambda_1-\Lambda_2\right)\right]$$

$$\eta = \frac{m_1 m_2}{\left(m_1+m_2\right)^2}$$

III. Comparison with LALInferenceMCMC



III. Comparison with LALInferenceMCMC

Parameter	LALInferenceMCMC	DeepHMC	JS
θ_{JN}	$2.6^{+0.4}_{-0.5}$	$2.5^{+0.4}_{-0.5}$	0.00200
ψ	$1.6^{+1.4}_{-1.5}$	$1.5^{+1.5}_{-1.4}$	0.00033
D_L	39^{+8}_{-14}	38^{+8}_{-15}	0.00166
z	$0.0087^{+0.0017}_{-0.0032}$	$0.0086^{+0.0017}_{-0.0032}$	0.00140
m_1	$1.47^{+0.15}_{-0.09}$	$1.49^{+0.14}_{-0.10}$	0.01848
m_2	$1.27^{+0.08}_{-0.11}$	$1.25^{+0.09}_{-0.11}$	0.01659
$\mathcal{M}$	$1.187^{+0.004}_{-0.002}$	$1.187^{+0.004}_{-0.002}$	0.00349
q	$0.87^{+0.12}_{-0.15}$	$0.84^{+0.12}_{-0.14}$	0.01718
δ	$-0.37^{+0.06}_{-0.06}$	$-0.37^{+0.06}_{-0.06}$	0.00142
α	$3.43^{+0.04}_{-0.04}$	$3.43^{+0.03}_{-0.04}$	0.00150
t_c	$1187008882.4312^{+0.0012}_{-0.0010}$	$1187008882.4311^{+0.0012}_{-0.0010}$	0.00264
χ_1	$0.003^{+0.028}_{-0.017}$	$0.004^{+0.027}_{-0.018}$	0.00123
χ_2	$0.002^{+0.028}_{-0.020}$	$0.003^{+0.028}_{-0.021}$	0.00118
χ_{eff}	$0.004^{+0.015}_{-0.010}$	$0.005^{+0.015}_{-0.010}$	0.00245
Λ_1	485^{+2096}_{-444}	255^{+770}_{-230}	0.05348
Λ_2	930^{+2347}_{-852}	446^{+1217}_{-400}	0.06730
$\tilde{\Lambda}$	762^{+1495}_{-543}	381^{+531}_{-237}	0.13416
$\delta\tilde{\Lambda}$	28^{+416}_{-502}	5^{+220}_{-208}	0.07847
ρ_{mf}	$32.47^{+0.09}_{-0.16}$	$32.28^{+0.09}_{-0.15}$	-
$\ln(\mathcal{L}_R)$	522^{+3}_{-5}	520^{+3}_{-6}	-

$$JS(P\|Q) = \frac{1}{2}D_{KL}(P\|Q) + \frac{1}{2}D_{KL}(Q\|P)$$

$$\sum_x P(x)\ln\left(\frac{P(x)}{Q(x)}\right)$$

$JS \leq 0.0015$

$JS \gtrsim 0.0015$, **manual inspection OK**

$JS > 0.0015$, **explained by Λ_i discrepancy**

III. Comparison with LALInferenceMCMC

Measured performances for N_{eff} = 5,000 SIS

Sampler	Wall time	CPU time	ACL_{max}	N_{eff}
LALInferenceMCMC	66.4 days	220.2 days	?	5,000 ?
DeepHMC	2.8 days	2.8 days	20	5,000

x80

- Machine used: Macbook Pro with 4 Intel i5-7360U cores, 2.3GHz clock rate
- LALInferenceMCMC ran on average on 3.3 CPUs vs 1 CPU for the DeepHMC

Conclusion

- DeepHMC reproduces LALInference's posterior distribution on GW170817.
- It is ~80 times faster (CPU time) to generate 5,000 SIS.
- DeepHMC's perspectives
  - Sample inclination's bimodality.
  - Precessing systems.
  - Parallelize Phase I to reduce wall time.
  - Validation on a broader range of systems (GW190412 with IMRPhenomHM, and GW190814 on-going).

Thank you !

Back up slides

- Define Hamiltonian

$$\mathscr{H}(q^\mu, p^\mu) = \mathscr{U}(q^\mu) + \mathscr{K}(p^\mu)$$

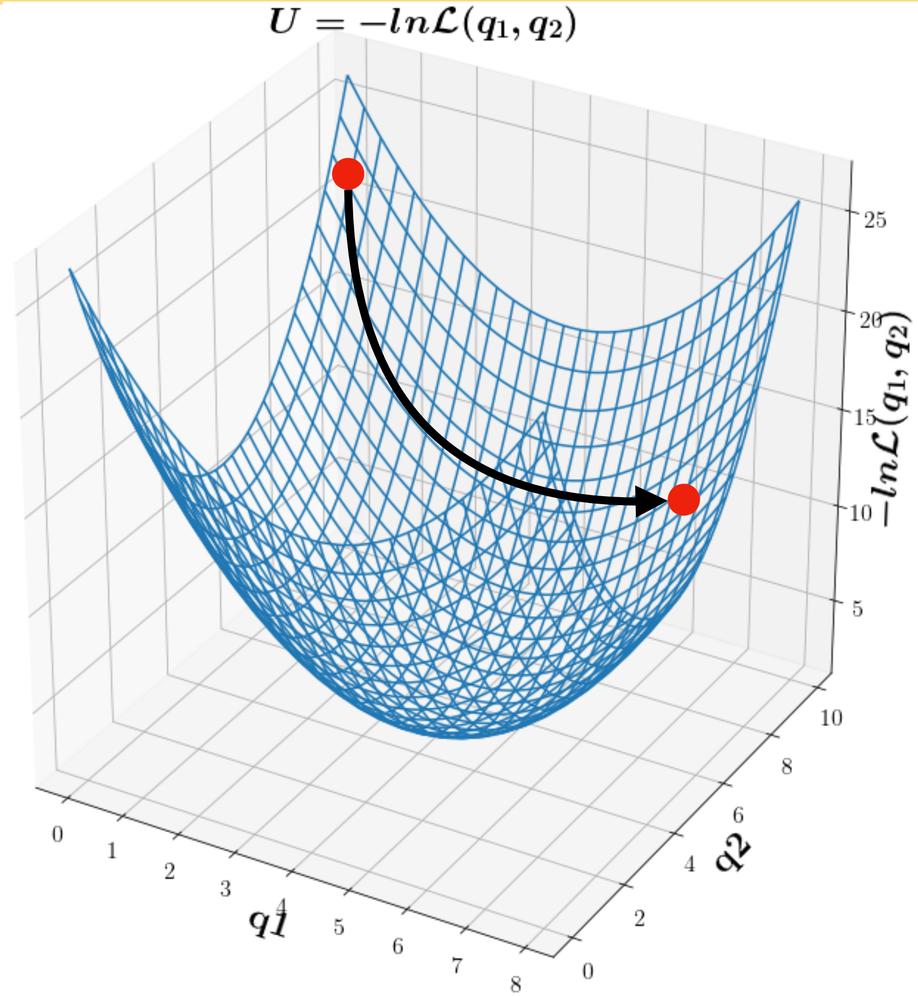
$$= -\,ln[\mathscr{L}(q^\mu)\pi(q^\mu)] + \frac{1}{2}M^{-1}_{\mu\nu}p^\mu p^\nu$$

- Canonical distribution

$$\Pi(q^\mu, p^\mu) \propto e^{-\mathscr{H}(q^\mu, p^\mu)}$$

$$\propto e^{-\mathscr{U}(q^\mu)}\; e^{-\mathscr{K}(p^\mu)}$$

$$\Pi(q^\mu, p^\mu) \propto \mathscr{L}(q^\mu)\pi(q^\mu)\; e^{-\frac{(p^\mu)^2}{2m^\mu}}$$



- If the momenta are drawn from a normal distribution, the marginal distribution for q^μ gives a sample set that asymptotically comes from the posterior distribution

Back up slides

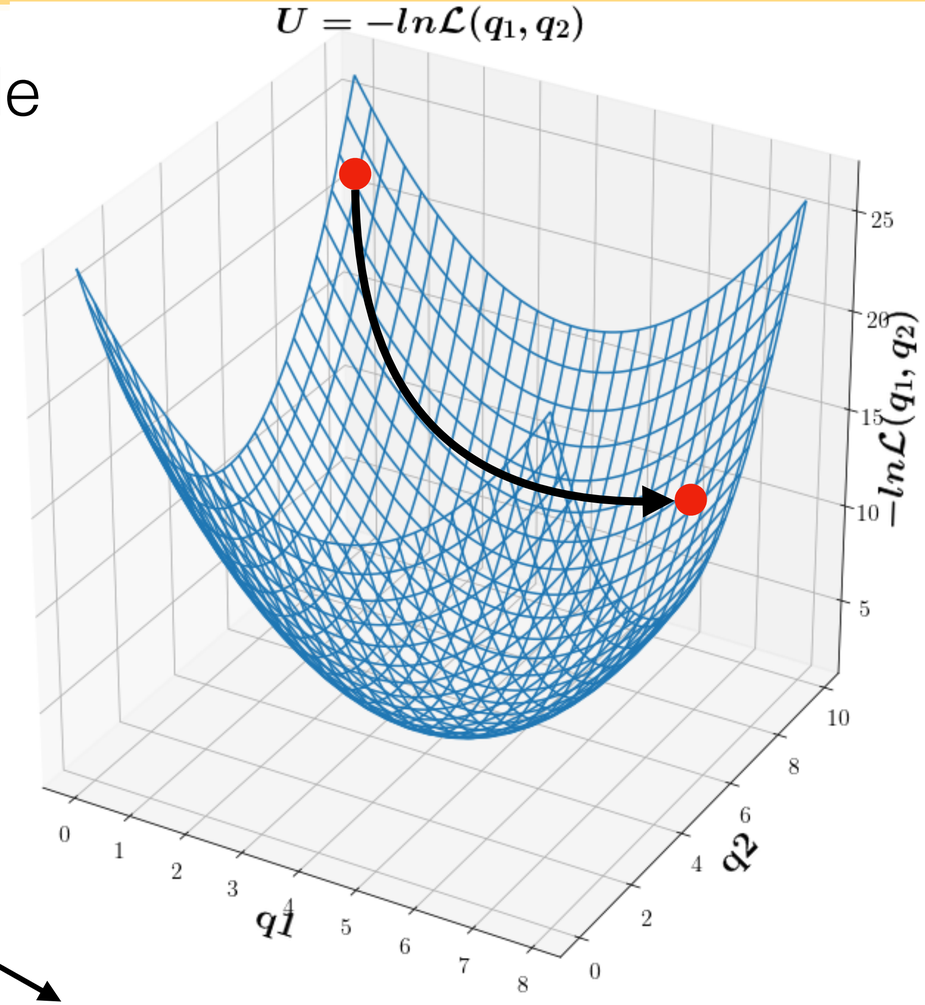
- Solve Hamilton's equations to get to the next sample

$$\frac{dq^\mu}{dt} = \frac{\partial \mathscr{H}}{\partial p^\mu}$$

$$\frac{dp^\mu}{dt} = -\frac{\partial \mathscr{H}}{\partial q^\mu}$$

- Allows for **distant state space proposals**

- Since the probability for the end point to be accepted depends on the conservation of $\mathscr{H}$, **the HMC has a high acceptance rate**



Small autocorrelation length

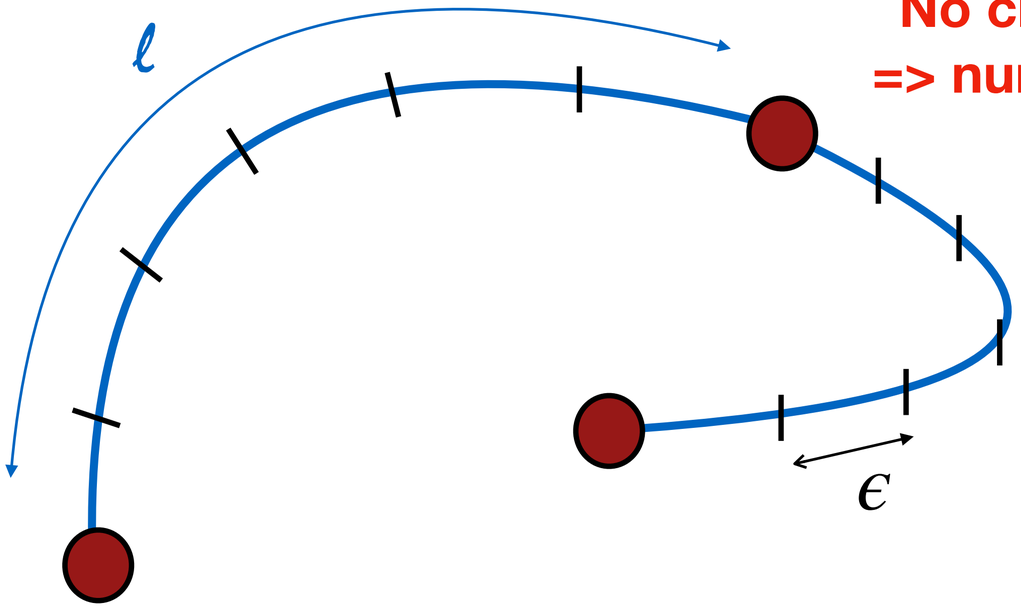
Back up slides

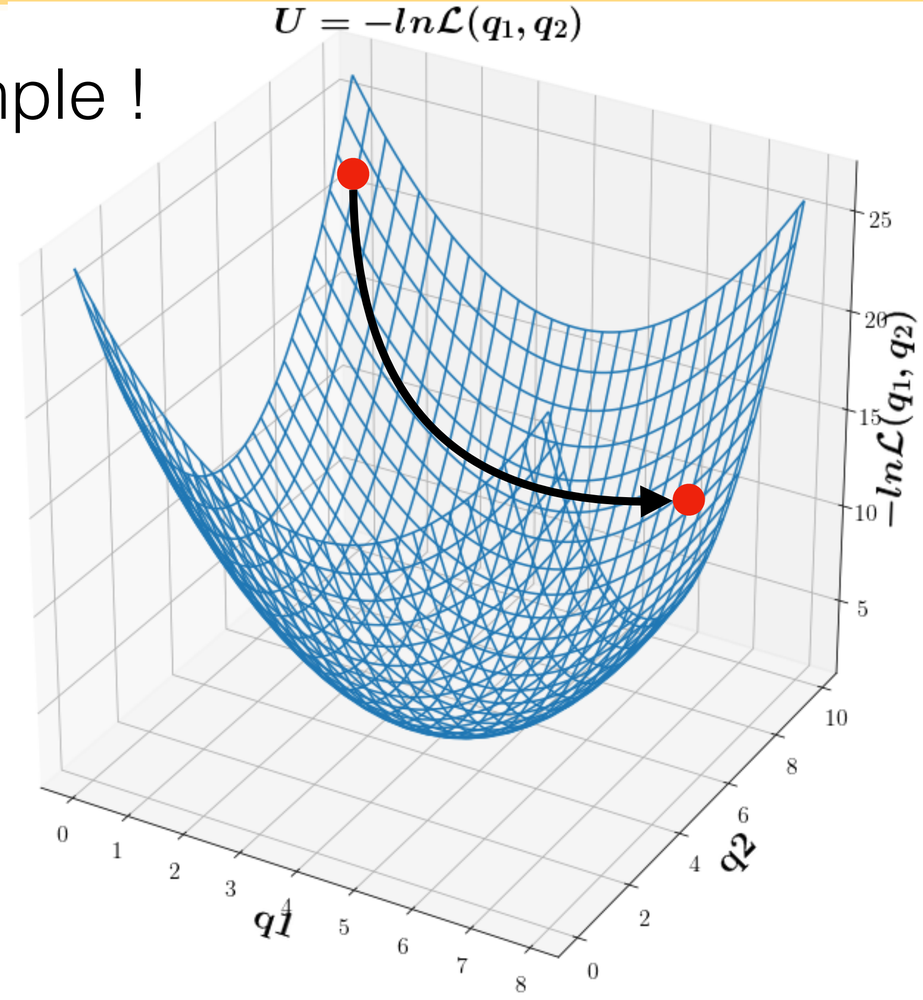
- Solve Hamilton's equations to get to the next sample !

$$\frac{dq^\mu}{dt} = \frac{\partial \mathscr{H}}{\partial p^\mu}$$

$$\frac{dp^\mu}{dt} = -\frac{\partial \mathscr{H}}{\partial q^\mu} \rightarrow \frac{\partial ln[\mathscr{L}(q^\mu)\pi(q^\mu)]}{\partial q^\mu}$$

No closed form solution
=> numerical differencing.





$\gtrsim Dl$ **waveforms to generate per trajectory.**

Back up slides

Set-up description

- Keras
- `Sequential()` model
- Loss function = `mean_squared_error`
- Optimizer = `Adam` (2014:Diederik K., Jimmy B.)
- Kernel initializer = `RandomNormal()`

Preprocessing of data

- Rescale q^μ and $\partial_\mu \ln \mathscr{L}$ such that they all have comparable ranges
- `sklearn.preprocessing.StandardScaler()`

Training

- Epochs = 60
- Batchsize = 128

Back up slides

Comparison with LALInferenceMCMC

Apples-to-apples comparison by

- running LAL on the same local machine
- same approximant
- same priors
- marginalization over phase
- no marginalization over calibration
- same starting point in parameter space

```
lalinference_mcmc
--amporder -1
--seglen 64.0
--srate 4096.0
--dont-dump-extras
--resume
--adapt-verbose
--approx IMRPhenomD_NRTidalpseudoFourPN
--fref 20
--no-detector-frame
--ifo H1
--ifo L1
--ifo V1
--H1-cache ../H-H1_LOSC_CLN_4_V1-1187007040-2048.lcf
--L1-cache ../L-L1_LOSC_CLN_4_V1-1187007040-2048.lcf
--V1-cache ../V-V1_LOSC_CLN_4_V1-1187007040-2048.lcf
--H1-psd ../GWTC1_GW170817_PSD_H1.dat
--L1-psd ../GWTC1_GW170817_PSD_L1.dat
--V1-psd ../GWTC1_GW170817_PSD_V1.dat
--H1-channel H1:LOSC-STRAIN
--L1-channel L1:LOSC-STRAIN
--V1-channel V1:LOSC-STRAIN
--H1-flow 30
--H1-fhigh 2047.9921875
--L1-flow 30
--L1-fhigh 2047.9921875
--V1-flow 30
--V1-fhigh 2047.9921875
--H1-timeslide 0
--L1-timeslide 0
--V1-timeslide 0
--trigtime 1187008882.45
--tempKill 1000000
--neff 5000
--ntemps 8
--adapt-temps
--outfile ../outdir/samples.hdf5
--randomseed 1870927565
--margphi
--chirpmass-min 1.1838325
--chirpmass-max 2.16847416667
--q-min 0.125
--comp-min 0.5
--comp-max 7.73105475907
--distance-max 75
--a_spin1-min -0.05
--a_spin1-max 0.05
--a_spin2-min -0.05
--a_spin2-max 0.05
--lambda1-min 0.0
--lambda1-max 5000.0
--lambda2-min 0.0
--lambda2-max 5000.0
--time 1187008882.43
--chirpmass 1.1976134456328078
--q 0.6415735002603987
--costheta_jn -0.45501295163890954
--rightascension 3.44616
--declination -0.408084
--polarisation 1.61012417485
--a_spin1 0.027842426346
--a_spin2 0.0278262466181
--lambda1 130.651548464
--lambda2 210.695175639
```

Back up slides

Extrapolation to `IMRPhenomPv2_NRTidal`, 128 sec, D = 12

~x20

Sampler	Nb CPUs	Wall time						CPU time
PBILBY	560	11 h						256.7 d
PBILBY (expected)	16	14.2 d						227.2 d
DEEPHMC (extrapolated)	1	12.6 d						12.6 d
		Phase I	Phase II	Phase III				
		10.2 d	1 h	Analytical 6.0 h	Hybrid 5.3 h	Numerical 45.0 h	DNN fit 1.3 h	

Expected comparison of the performances of DeepHMC with PBilby if analysing 128s of GW170817 real data sampled at 4096Hz with the IMRPhenomP_NRTidal approximant while marginalizing over (ϕ_c, t_c, D_L) . The 1st line reports performances of PBilby from the analysis used in Romero-Shaw et al, the 2nd line infers its performances if the parallelization uses 16 cores instead of 560, the 3rd line extrapolates the performances of DeepHMC from its aligned-spin results analysis and details how the computational cost would spread over the different phases and trajectory types and on retraining the DNN.

Back up slides

- Trajectories in phase space are found solving Hamilton's equations

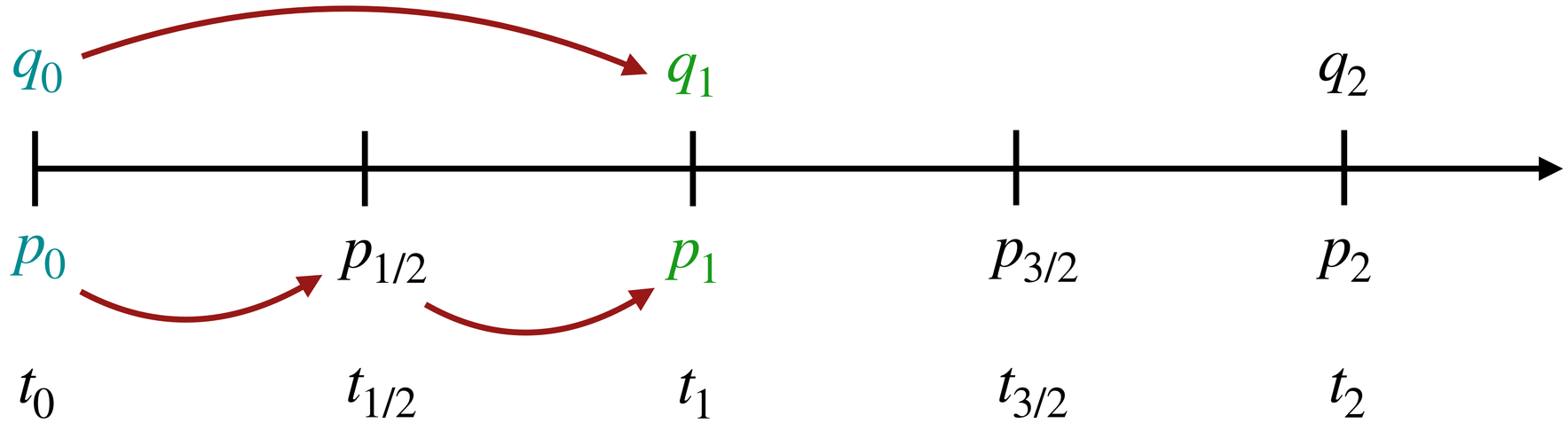
$$\begin{cases} \dfrac{dp^\mu}{dt} = -\dfrac{\partial \mathcal{U}}{\partial q^\mu} \\ \dfrac{dq^\mu}{dt} = \dfrac{\partial \mathcal{K}}{\partial p^\mu} \end{cases} \longrightarrow \begin{cases} \dfrac{dp^\mu}{dt} = \dfrac{\partial ln[\mathcal{L}(q^\mu)\pi(q^\mu)]}{\partial q^\mu} \\ \dfrac{dq^\mu}{dt} = M^{-1}_{\mu\nu} p^\nu = (m^\mu)^{-1} p^\mu \end{cases}$$

- Can't be solved analytically

- We need an integrator that
  - Conserves the Hamiltonian
  - Is time reversible
  - Conserves phase space volume

Leap frog algorithm

Back up slides



1/2 step in momenta

$$p^\mu\left(t+\frac{\epsilon}{2}\right) = p^\mu(t) + \frac{\epsilon}{2}\frac{\partial ln[\mathscr{L}(q^\mu)\pi(q^\mu)]}{\partial q^\mu}\Bigg|_{q^\mu(t)}$$

Full step in position

$$q^\mu(t+\epsilon) = q^\mu(t) + \frac{\epsilon}{m^\mu}p^\mu\left(t+\frac{\epsilon}{2}\right)$$

1/2 step in momenta

$$p^\mu(t+\epsilon) = p^\mu\left(t+\frac{\epsilon}{2}\right) + \frac{\epsilon}{2}\frac{\partial ln[\mathscr{L}(q^\mu)\pi(q^\mu)]}{\partial q^\mu}\Bigg|_{q^\mu(t+\epsilon)}$$

Back up slides

Scaled leapfrog

$$\epsilon^\mu = s^\mu \epsilon$$

$$s^\mu = m_\mu{}^{-1/2}$$

$$\tilde{p}^\mu = s^\mu p^\mu$$

1/2 step in momenta

$$\tilde{p}^\mu\left(t + \frac{\epsilon^\mu}{2}\right) = \tilde{p}^\mu(t) + \frac{\epsilon^\mu}{2}\frac{\partial ln[\mathscr{L}(q^\mu)\pi(q^\mu)]}{\partial q^\mu}\Bigg|_{q^\mu(t)}$$

Full step in position

$$q^\mu(t + \epsilon^\mu) = q^\mu(t) + \epsilon^\mu \tilde{p}^\mu\left(t + \frac{\epsilon^\mu}{2}\right)$$

1/2 step in momenta

$$\tilde{p}^\mu(t + \epsilon^\mu) = \tilde{p}^\mu\left(t + \frac{\epsilon^\mu}{2}\right) + \frac{\epsilon^\mu}{2}\frac{\partial ln[\mathscr{L}(q^\mu)\pi(q^\mu)]}{\partial q^\mu}\Bigg|_{q^\mu(t+\epsilon^\mu)}$$

Operation	Derivative method	$\tilde{h}_s$	$\tilde{h}_{ifo}(t_c = 0)$	$e^{-i2\pi f \delta t_c^{ifo}}$	$\tilde{h}_{ifo}(t_c = 0) \times e^{-i2\pi f \delta t_c^{ifo}}$	$\langle f \vert g \rangle$
$\ln \mathcal{L}$	-	1	n_{ifo}	n_{ifo}	n_{ifo}	$2\, n_{ifo}$
$\partial_{\cos\theta_{JN}} \ln \mathcal{L}$	Forward	1	n_{ifo}	0	n_{ifo}	$2\, n_{ifo}$
$\partial_{\psi} \ln \mathcal{L}$	Forward	0	n_{ifo}	0	n_{ifo}	$2\, n_{ifo}$
$\partial_{\ln D_L} \ln \mathcal{L}$	Forward	0	n_{ifo}	0	n_{ifo}	$2\, n_{ifo}$
$\partial_{\ln \mathcal{M}} \ln \mathcal{L}$	Central	2	$2\, n_{ifo}$	0	$2\, n_{ifo}$	$3\, n_{ifo}$
$\partial_{\ln \mu} \ln \mathcal{L}$	Central	2	$2\, n_{ifo}$	0	$2\, n_{ifo}$	$3\, n_{ifo}$
$\partial_{\sin(\delta)} \ln \mathcal{L}$	Forward	0	n_{ifo}	n_{ifo}	n_{ifo}	$2\, n_{ifo}$
$\partial_{\alpha} \ln \mathcal{L}$	Forward	0	n_{ifo}	n_{ifo}	n_{ifo}	$2\, n_{ifo}$
$\partial_{\ln \delta t_c} \ln \mathcal{L}$	Central	0	$2\, n_{ifo}$	$2\, n_{ifo}$	$2\, n_{ifo}$	$3\, n_{ifo}$
$\partial_{\chi_1} \ln \mathcal{L}$	Central	2	$2\, n_{ifo}$	0	$2\, n_{ifo}$	$3\, n_{ifo}$
$\partial_{\chi_2} \ln \mathcal{L}$	Central	2	$2\, n_{ifo}$	0	$2\, n_{ifo}$	$3\, n_{ifo}$
$\partial_{\Lambda_1} \ln \mathcal{L}$	Forward	1	n_{ifo}	0	n_{ifo}	$2\, n_{ifo}$
$\partial_{\Lambda_2} \ln \mathcal{L}$	Forward	1	n_{ifo}	0	n_{ifo}	$2\, n_{ifo}$
TOTAL	-	12	$18\, n_{ifo}$	$5\, n_{ifo}$	$18\, n_{ifo}$	$31\, n_{ifo}$
Time per operation	-	25.4 ms	1.0 ms	3.5 ms	1.6 ms	0.9 ms
TOTAL time	**581 ms**	304.8 ms	54.0 ms	52.5 ms	86.4 ms	78.3 ms

Based on 59s of data sampled at 4096 Hz using the IMRPhenomD_NRTidal approximant, we list the number of operations required to generate the log-likelihood, plus the 12 gradients of the loglikelihood in a single step of the leapfrog evolution. Here, columns three to seven represent the source frame waveform, the waveform projected onto each detector with $t_c = 0$, the time shift at each detector, the time shifted waveform at each detector, and the noise weighted inner product respectively. In each column, n_{ifo} is the number of detectors used in the analysis. The second last line gives the time per single calculation of each operation, while in the last line we calculate a total time of 581ms for the generation of the log-likelihood and its 8 gradients, as well as a breakdown of the total cost to detail where the main contributions are coming from.