Emulation and Assessment of Gradient-Based Samplers in Cosmology

Arrykrishna Mootoovaloo [*]
Department of Astrophysics
University of Oxford
Keble Road, Oxford, OX1 3RH, UK

Carlos García-García
Department of Astrophysics
University of Oxford
Keble Road, Oxford, OX1 3RH, UK

David Alonso
Department of Astrophysics
University of Oxford
Keble Road, Oxford, OX1 3RH, UK

Jaime Ruiz-Zapatero
Department of Astrophysics
University of Oxford
Keble Road, Oxford, OX1 3RH, UK

Abstract

We assess gradient-based samplers like the No-U-Turn Sampler (`NUTS`) compared to traditional Metropolis-Hastings algorithms in tomographic 3×2 point analyses using DES Year 1 data and a simulated LSST-like survey. These studies involve 20 and 32 nuisance parameters, respectively. We implement a differentiable forward model using $\mathtt{JAX} - \mathtt{COSMO}$ and derive parameter constraints using `NUTS` and Metropolis-Hastings algorithms. `NUTS` shows a relative efficiency gain of $\mathcal{O}(10)$ in terms of effective samples per likelihood evaluation but only a factor of ~ 2 in terms of computational time due to the higher gradient computation cost. We validate these results with analytical multivariate distributions, concluding that `NUTS` can be beneficial for sampling high-dimensional parameter spaces in Cosmology, though the efficiency gain is modest for moderate dimensions ($\mathcal{O}(50)$).

1 Introduction

In this paper, we explore the significance of differentiability in sampling both cosmological and nuisance parameters within standard cosmological analyses. To enable this, we develop a differentiable linear matter power spectrum emulator. Emulation involves building an approximate function that can accurately model the desired quantity. The idea of emulation is in fact an old concept. For instance, Eisenstein & Hu (1998) derived an analytic expression to describe the linear matter transfer function in the presence of cold dark matter, radiation, and baryons. However, deriving these types of expressions in general, purely in terms of cosmological parameters, requires significant human ingenuity, particularly in the presence of growing model complexity and ever more stringent accuracy requirements. Various types of emulators have been designed, with their own advantages and disadvantages. Techniques such as polynomial regression, neural networks, Gaussian Processes (GPs) and genetic algorithms have been explored by different groups. For example, Fendt & Wandelt (2007) used polynomial regression to emulate the CMB power spectra, while Habib et al. (2007) used Gaussian Processes, together with a compression scheme, to emulate the non-linear matter spectrum from simulations. Recently, Aricò et al. (2021); Spurio Mancini et al. (2022); Bonici et al. (2024) developed a neural network framework to emulate different power spectra. Moreover, Bartlett et al. (2023, 2024) used Symbolic Regression – a technique for finding mathematical expression of the function of interest – to emulate the matter power spectrum.

[*] `arrykrishna.mootoovaloo@physics.ox.ac.uk`

Machine Learning and the Physical Sciences Workshop, NeurIPS 2024.

Our contributions in this work are as follows:

1. we integrate an emulator for the linear matter power spectrum in $\texttt{JAX}-\texttt{COSMO}$ (Campagne et al. 2023) and leverage its existing functionalities for computing power spectra for galaxy clustering and cosmic shear,
2. we take advantage of gradient-based samplers such as NUTS to sample the posterior of the cosmological and nuisance parameters using DES Year 1 data (Abbott et al. 2018) and a future LSST-like survey data and
3. we perform an in-depth assessment of whether differentiability is helpful in this context.

2 Method

Our emulator is based on Gaussian Process. We decompose the linear matter power spectrum in two parts:

$$P_l(k, z, \boldsymbol{\theta}) = G(z, \boldsymbol{\theta})\, P_l^0(k, \boldsymbol{\theta}) \tag{1}$$

where P_l^0 is the linear matter power spectrum evaluated at redshift, $z = 0$. $\boldsymbol{\theta}$ is the vector of input cosmological parameters, $\boldsymbol{\theta} = \{\sigma_8,\, \Omega_c,\, \Omega_b,\, h,\, n_s\}$. The input training points are generated using Latin Hypercube Sampling (LHS) which ensures the points randomly cover the full space. In particular, the emulator is built over the redshift range of $z \in [0.0,\, 3.0]$ and the wavenumber range of $k \in [10^{-4},\, 50]$ in units of Mpc^{-1}. We use $N_\theta = 1000$ training points. In particular, we record the targets $(G$ and $P_l^0)$ over $N_z = 20$ redshift values, equally spaced in linear scale for the redshift range and $N_k = 30$ wavenumber values, equally spaced in logarithmic scale for the wavenumber range. This gives us two training sets, $\mathsf{Y}_k \in \mathbb{R}^{N_\theta \times N_k}$ and $\mathsf{Y}_G \in \mathbb{R}^{N_\theta \times N_z}$. We then build 50 independent models as a function of the cosmological parameters. Once trained and stored, they are coupled to the $\texttt{JAX}-\texttt{COSMO}$ code, which is then use to compute weak lensing and galaxy clustering power spectra.

To sample the posterior, we use the gradient-based NUTS algorithm implemented in `numpyro` (Phan et al. 2019; Bingham et al. 2019). We compare its performance with the standard Metropolis-Hastings algorithm in `Cobaya`, which does not rely on gradient information (Lewis 2013). Throughout, we compute different metrics, such as the sampling efficiency as well as the Gelman-Rubin statistics to assess the efficiency of the chain and the convergence respectively (Gelman & Rubin 1992).

3 Data

We analyse two cosmological datasets: DES Y1, with 405 data points and 20 nuisance parameters, and a future LSST-like dataset, with 903 data points and 32 nuisance parameters, following necessary scale cuts. These nuisance parameters are integral to the cosmological modelling framework, accounting for astrophysical and systematic effects such as intrinsic alignment, biases, and shifts in the tomographic bins.

Besides sampling the posterior for the two cosmological datasets, we also investigate two multivariate analytical functions: the multivariate normal distribution and the Rosenbrock function. In the cosmological analyses, the parameter set is fixed, whereas, for the analytical functions, we evaluate the metrics across increasing dimensionality.

4 Result

In Figure 1, we show the accuracy of the emulator, which was then embedded into the $\texttt{JAX}-\texttt{COSMO}$ pipeline. For the range of redshifts and wavenumber considered and the domain of the cosmological parameters, the quantities P_l and G are accurate up to $\leq 1\%$ and $\leq 0.01\%$ respectively. Recall that we are using 1000 LH samples to build the emulator. Generating the 1000 training points using the simulator, CLASS (Lesgourgues 2011) took around 1 hour while training the GPs took around 2 hours on a desktop computer. The training of GPs is expensive because of the $\mathcal{O}(N^3)$ cost in the Cholesky decomposition. However, once they are trained and stored, prediction is very fast and computing the log-likelihood is of the order of milliseconds. Moreover, one can use the fixed $\boldsymbol{\alpha}$s

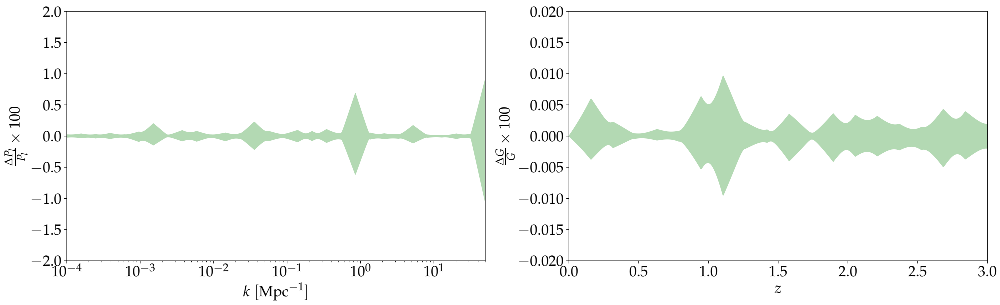


Figure 1: The left panel shows the accuracy for the linear matter power spectrum, evaluated at $z = 0$ over a wavenumber range of $k \in [10^{-4},\, 50]$ in units of Mpc^{-1} and the right panel shows the accuracy for the quantity, G evaluated over the redshift range of $z \in [0.0\, 3.0]$. These quantities can be robustly calculated within an accuracy of 1%.

and the kernel pre-trained hyperparameters in any GP implementation irrespective of whether we use `numpy`, `pyTorch`, `TensorFlow` and `JAX`. Moreover, the priors are sufficiently broad that the emulation framework can be used for different probes, for example, weak lensing as in this context.

Figure 2 compares the marginalised 1D and 2D distributions of the cosmological parameters using the emulator with `Cobaya` and `NUTS`. Under this configuration, the potential scale reduction factor is equal to 1.00 for all parameters. Sampling the posterior with `NUTS` takes $\sim$ 13 hours for two chains with `numpyro` using a single GPU. Alternatively, a single run using `Cobaya`, whether with the emulator or EH in $\texttt{JAX} - \texttt{COSMO}$, takes approximately 5 hours to sample the posterior. Note that the chains generated by both samplers did not contain the same number of samplers, and therefore the difference in time above is not reflective of their relative performance.

Moreover, in order to quantify the difference between the inferred parameters with either sampler, we use the "difference of Gaussians" statistic:

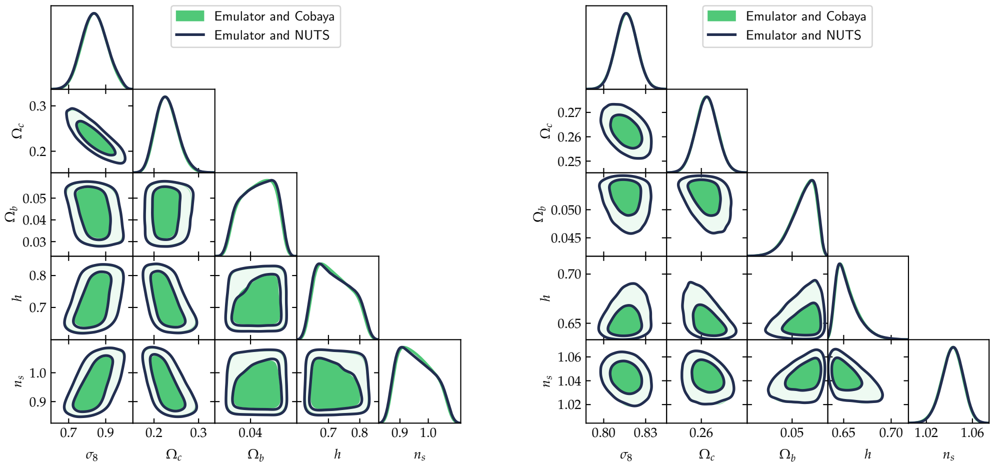
$$\delta = \frac{|\mu_{\texttt{NUTS}} - \mu_{\texttt{Cobaya}}|}{\sqrt{\sigma^2_{\texttt{NUTS}} + \sigma^2_{\texttt{Cobaya}}}}. \tag{2}$$

The maximum difference among the set of parameters considered in this experiment is ~ 0.1. We also compute the average of the scaled effective sample size, N_{eff}, to compare the samplers. We define the efficiency gain as:

$$\gamma = \frac{N_{\text{eff, }\texttt{NUTS}}}{N_{\text{eff, }\texttt{Cobaya}}}. \tag{3}$$

The relative gain in efficiency when using `NUTS` compared to `Cobaya` is $\mathcal{O}(10)$. When using `Limberjack` and `NUTS` in `Turing.jl`, Ruiz-Zapatero et al. (2023) estimated a gain in N_{eff} of ~ 1.7, compared to the samples obtained using Metropolis-Hastings implemented in `Cobaya` (García-García et al. 2021). In addition, when using reverse mode automatic differentiation (the default setup in `numpyro`), the cost of a single gradient calculation to the cost of a single likelihood evaluation is ~ 4.5 (either with the emulator or EH). The gain in efficiency is better compared to the cost of the gradient evaluation. With `julia`, Ruiz-Zapatero et al. (2023) found this ratio to be ~ 5.5 when using forward mode automatic differentiation. Note that the differences in the values above can be attributed to the fact that different samplers will, in general, have different implementations. Taking into account the more expensive gradient evaluation, we find that the overall efficiency gain of `NUTS` with respect to `Cobaya`, when measured in terms of computing time on the same platform, is ~ 2.

In both the multivariate normal distribution and the Rosenbrock function examples, the potential scale reduction factor is close to one when either `NUTS` or `Cobaya` is used to sample the function. However, with a tricky function such as the Rosenbrock function, we find that the potential scale reduction factor gets worse ($R \sim 1.0 - 1.4$) with `Cobaya` as the dimensionality increases. Moreover, the acceptance probability when `NUTS` is used is always $\gtrsim 0.7$ with either the multivariate normal



(a) DES Analysis

(b) Analysis with LSST-like data

Figure 2: The 1D and 2D marginalised posterior distribution of all the cosmological parameters. The green contours show the distribution when the emulator is used with the Cobaya sampler while the solid black curves correspond to the setup where NUTS is used for sampling the posterior distribution. There is negligible difference in the posterior when comparing Cobaya and NUTS. The left panel shows the posterior obtained when using the DES data while the right panel shows the contours obtained with the simulated LSST-like data.

or the Rosenbrock function. On the other hand, Cobaya has an acceptance probability of ~ 0.3 when sampling the multivariate normal distribution. With the Rosenbrock function, the acceptance probability varies from ~ 0.17 to ~ 0.1 as the dimensionality increases.

Based on the experiments performed with the analytical, we find that NUTS always produces more effective samples, irrespective of the function employed. With the Rosenbrock function depicting non-Gaussianity – characterized by its non-linear and asymmetric shape – it is expected that samplers will result in a reduction of N_{eff}. For $d \leq 100$, both samplers are able to recover the correct shape of the posterior distribution. However, NUTS is more likely to scale better to higher dimensions $(d > 100)$ as a result of its consistent high N_{eff}.

5 Challenges and Future Work

A notable challenge we faced was ensuring compatibility between CUDA and JAX. In this work, we used CUDA 11 and JAX version 0.3.25, but updating to the latest versions of CUDA and JAX could introduce complications. From a scientific standpoint, we aim to identify a problem with dimensions $(d > 100)$ where the methodology described here can highlight the significance of gradient-based samplers.

6 Conclusion

In this work, we have performed a quantitative assessment of different aspects related to emulation and gradient-based samplers. We integrated a linear matter power spectrum emulator into $\texttt{JAX} - \texttt{COSMO}$, achieving $\sim 1\%$ accuracy with just 1000 training points. This is different with deep learning frameworks that require significantly more data for similar accuracy. Constraints from NUTS and Cobaya samplers show strong agreement. In the DES analysis, NUTS achieved a 10-fold efficiency gain for a 25-parameter system, though this reduces to a factor of 2 when accounting for gradient computation costs. NUTS excels in high-dimensional spaces $(d > 100)$, showing better convergence and effective sample size than Cobaya. For a 37-dimensional LSST-like survey, NUTS is twice as effective as Cobaya. An in-depth examination using the Rosenbrock function confirms the sampling efficiency gain of NUTS. This suggests that NUTS is not only advantageous in terms of convergence and effective sample size but also provides improved exploration of complex, non-trivial functions. Overall, these findings highlight the contexts where NUTS outperforms traditional non-gradient based samplers, making it a valuable tool for Bayesian inference in a wide range of applications.

References

Abbott, T. M. C., Abdalla, F. B., Alarcon, A., et al. 2018, , 98, 043526

Aricò, G., Angulo, R. E., & Zennaro, M. 2021, arXiv e-prints, arXiv:2104.14568

Bartlett, D. J., Kammerer, L., Kronberger, G., et al. 2023, arXiv e-prints, arXiv:2311.15865

Bartlett, D. J., Wandelt, B. D., Zennaro, M., Ferreira, P. G., & Desmond, H. 2024, , 686, A150

Bingham, E., Chen, J. P., Jankowiak, M., et al. 2019, J. Mach. Learn. Res., 20, 20

Bonici, M., Bianchini, F., & Ruiz-Zapatero, J. 2024, The Open Journal of Astrophysics, 7, 10

Campagne, J.-E., Lanusse, F., Zuntz, J., et al. 2023, The Open Journal of Astrophysics, 6, 15

Eisenstein, D. J. & Hu, W. 1998, , 496, 605

Fendt, W. A. & Wandelt, B. D. 2007, , 654, 2

García-García, C., Ruiz-Zapatero, J., Alonso, D., et al. 2021, , 2021, 030

Gelman, A. & Rubin, D. B. 1992, Statistical Science, 7, 457

Habib, S., Heitmann, K., Higdon, D., Nakhleh, C., & Williams, B. 2007, , 76, 083503

Lesgourgues, J. 2011, arXiv e-prints, arXiv:1104.2932

Lewis, A. 2013, , 87, 103529

Phan, D., Pradhan, N., & Jankowiak, M. 2019, arXiv preprint arXiv:1912.11554

Ruiz-Zapatero, J., Alonso, D., García-García, C., et al. 2023, arXiv e-prints, arXiv:2310.08306

Spurio Mancini, A., Piras, D., Alsing, J., Joachimi, B., & Hobson, M. P. 2022, , 511, 1771