The Use of Differentiable Physics Simulation Techniques in Cosmology

Kaylie Hausknecht

April 25, 2023

1 Introduction

The field of data-driven Cosmology is centered around solving a wide variety of inverse problems. For instance, there is a wide class of problems where given a cosmological observable we may seek to identify a model and the associated parameters that have the maximum likelihood of producing the data. Given a set of model parameters that we desire to learn about through observations, another class of inverse problems in cosmology is to design a probe that is optimized for constraining those parameters and eliminating degeneracies. In other cases, inverse problems in cosmology can revolve around data processing and data reconstruction. In general, solving these types of inverse problems requires running parameter sensitivity analyses, solving optimization problems, or using high-dimensional inference techniques. Importantly, many of the most useful models in cosmology cannot be expressed analytically and require numerical solutions and simulations to be used (i.e. Boltzmann solvers, N-body simulations, hydrodynamical simulations, and growth of structure simulations). The computational complexity of these simulations and non-analytical nature means that computing gradients with respect to these models is not straightforward and can require approximations that may be numerically unstable or inaccurate in some cases.

Over the past few decades, the field of machine learning has made extraordinary improvements in algorithms used for automatic differentiation. Over the past several years, machine learning methods have begun to trickle into physics. However, because of the black-box nature of most machine learning models and the difficulty of enforcing physical constraints with them, I believe that the most impactful piece of the machine learning revolution for physics will be algorithms for automatic differentiation and not neural networks. In particular, differentiable programming methodologies that have emerged out of machine learning libraries can be utilized for programming physics simulations.

Differentiable physics simulation packages in the fields of molecular dynamics [1] and computational fluid dynamics [2] have been written using JAX. JAX is a Python library whose development

was largely spearheaded by Google. JAX was intended to transform popular Python functions and packages into a form that makes them automatically differentiable [3]. JAX uses Jacobian-Vector Products and Vector-Jacobian Products to power its automatic differentiation algorithms. The JAX team has successfully converted most NumPy and SciPy functions into forms that are differentiable, and separate JAX packages have developed more specific functionalities for specific domains, computational fluid dynamics and molecular dynamics being two examples. Programming differentiable physics simulations requires us to write our code in clever and sometimes very particular ways, but it does not require us to sacrifice the laws of physics or simplify the simulations themselves. These simulations can take time to ensure they are fully differentiable but this time investment can prove useful in different fields because of the opportunities enabled by having fully differentiable simulations.

Recently, the Differentiable Universe Initiative has begun to develop a differentiable cosmology library called JAX-Cosmo [4]. In their paper, Campagne et al. compare the functionality of JAX-Cosmo to the Core Cosmology Library. They demonstrate the usefulness of JAX by calculating fisher matrices to estimate the constraints that weak lensing probes can put on cosmological parameters, and they use Hamiltonian Monte Carlo sampling techniques to generate the posterior probability distributions for the $\Lambda - CDM$ cosmological parameters based on data from the Dark Energy Survey Year 1 lensing and clustering data. In addition to the use cases explored in the original JAX-Cosmo paper, translating commonly used models and simulations in cosmology into a form that is end-to-end differentiable has the potential to improve inference, allow us to solve optimization problem, calculate complex parameter sensitivities, and improve any calculations that require gradient information.

In this project, I explore several application areas of differentiable cosmology. Importantly, JAX-Cosmo has not implemented a differentiable Boltzmann solver, although this has been done in Julia which I spent some time trying to use. This limited the complexity of the examples I was able to work with, so I mostly focused on supernovae cosmology and in Section 5 the matter power spectrum. Nevertheless, even in the case of Type 1a Supernovae, the efficiency improvements and exciting capabilities that we get by translating our cosmological models into a form that is differentiable are clear.

The breakdown of my project and paper is as follows. In Section 2, I describe writing a model of the distance modulus for Type 1a Supernovae in a way that is differentiable and then using a Hamiltonian Monte Carlo sampler to fit cosmological parameters using data from the Pantheon+SH0es Catalogs. In Section 3, I use JAX to compute Fisher forecasts for a supernovae probe and compare the Fisher forecast to the posteriors found in the previous section. In Section 4, I design and show that we can solve an optimization problem for maximizing the constraints that we can put on the dark energy equation of state parameters. Finally in Section 5, I show how we can compute derivatives of the matter power spectrum with respect to different cosmological parameters. I link the Python notebooks

that I wrote the code for this project in in Google Colab. Throughout the notebook, I time many of the computations in JAX to show the efficiency improvements we get from writing differentiable cosmological models.

2 Hamiltonian Monte Carlo on Type 1a Supernovae Data

Type 1a supernovae can occur in White Dwarf stars that are a part of stellar binaries. In these cases, if the other star in the binary accretes mass onto the white dwarf, eventually the electron degeneracy pressure within the star becomes insufficient for counteracting the gravitational collapse of the star due to the accreting mass. The point at which electron degeneracy pressure becomes too weak to counteract collapse occurs at a theoretically prescribed critical mass called the Chandrashekar mass. When the white dwarf collapses, it causes a supernova explosion with a signature light curve. Since we expect this collapse occur at the same critical mass for all white dwarf stars, since this mass is set by the physics of degenerate fermi gases and not the particulars of the stellar binary, we expect each supernovae explosion to have the same intrinsic luminosity. As such, Type 1a supernovae provide an important laboratory for probing cosmological distances because of their known luminosities. Accordingly, they are often referred to as "standard candles."

We define the distance modulus μ as

$$\mu = 5 \log_{10} \left[\frac{D_L}{\text{Mpc}} \right] + 25 = m - M \tag{1}$$

where m is the apparent magnitude, or apparent brightness, and M is the absolute magnitude, or intrinsic brightness. Since we expect M to be roughly the same for all Type 1a supernovae, we can find the distance modulus for a given Type 1a Supernovae just based on its apparent magnitude. The distance modulus can then be related to and used to constrain cosmological parameters through D_L, the luminosity distance, which it can also be written in terms of. The luminosity distance is given by

$$D_L = (1+z) \int_z^\infty \frac{c}{H(z')} dz' \tag{2}$$

where the Hubble parameter as a function of redshift is given by

$$H(z) = H_0 \sqrt{\Omega_r (1+z)^4 + \Omega_m (1+z)^3 + \Omega_K (1+z)^2 + \Omega_\Lambda}. \tag{3}$$

As such, measuring the distance modulus for Type 1a Supernovae data indirectly probes cosmological parameters, although it is more constraining for some parameters than others.

For this project, we use Type 1a Supernovae data from the Pantheon+ and SH0ES (Supernovae and H0 for the Equation of State of dark energy) Collaboration [5, 6]. This dataset includes samples of 1701 Type 1a Supernovae, whose distance moduli are plotted in Figure 1.

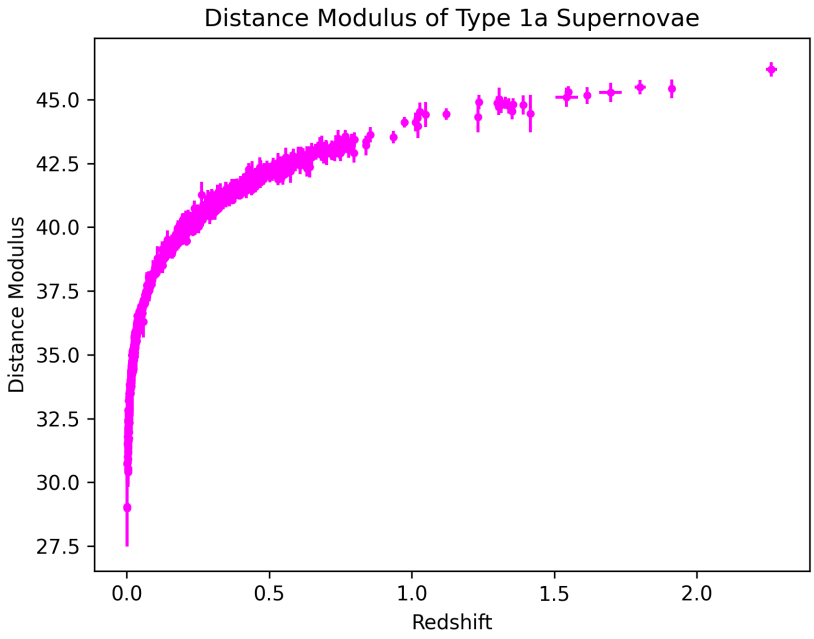

Figure 1: The distance modulus for 1701 Type 1a Supernovae from the Pantheon+SH0es collaboration.

The goal of this part of my project was to write a differentiable function that can compute μ for an object given a redshift and cosmological parameters. In the attached code files, I was able to do this by using functions from JAX NumPy and JAX-Cosmo's implementation of cosmological distance calculations. I tested and demonstrated the differentiability of this implementation of by end-to-end differentiating through the model using JAX's grad function. Figure 2 shows the result of differentiating $\mu(z, \vec{p})$, where $\vec{p}$ represents input cosmological parameters with respect to w_0 and w_a.

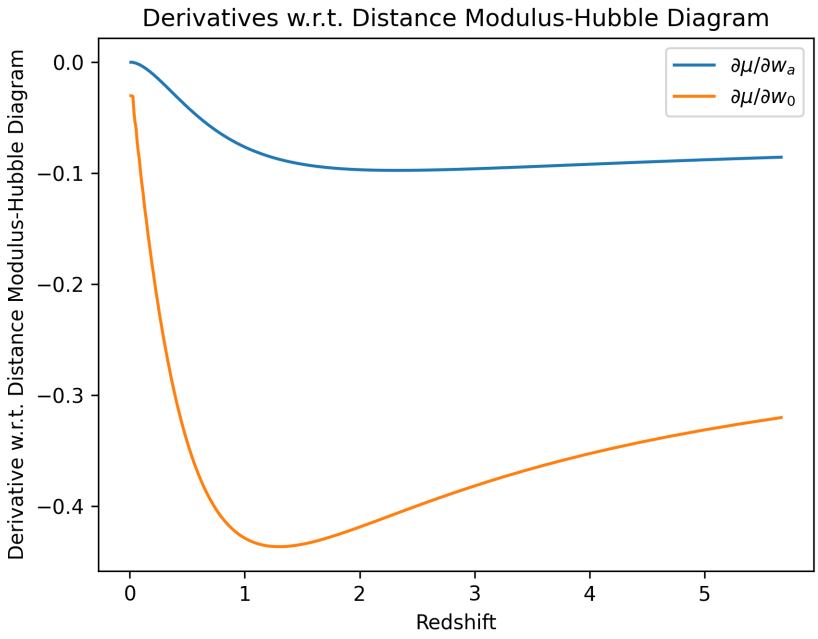

Figure 2: Computing derivatives through model of cosmological distance modulus with respect to dark energy equation of state parameters.

Here, w_0 and w_a are parameters related to the dark energy equation of state [7]. They affect the density of dark energy in the following way

$$w(a) = w_0 + w_a(1 - a) \tag{4}$$

where

$$\rho_{\text{DE, a}} = \rho_{\text{DE, 0}} \exp\left[-3 \int_1^a (1 + w(a'))d \ln a' \right]. \tag{5}$$

In Λ-CDM, the fiducial value of $w_0 = -1$ and $w_a = 0$. Based on Figure 2, we see that the distance modulus model is most sensitive to the dark energy equation of state parameters around a redshift of 1 (indicated by the derivatives with the greatest absolute magnitude). This checks with intuition because this is the period leading up to the dark energy-dominated regime that we are now in at $z = 0$. The ability to differentiate end-to-end through models will be discussed further in section 5, but for now, we can take this as proof-of-concept that the implementation of the distance modulus calculation was fully differentiable.

With this differentiable model of the distance modulus, my goal was to perform Hamiltonian Monte Carlo sampling to find the posterior probability distributions for cosmological parameters based on the supernovae dataset. The differentiability of the model is important here because Hamiltonian Monte Carlo methods, and in particular the No-U-Turn Sampler that I will use here, are gradient-based

inference techniques [8]. Hamiltonian Monte Carlo techniques overcome some of the known limitations of traditional Markov Chain Monte Carlo techniques such as the popular Metropolis Hastings Algorithm. These limitations include being inefficient at sampling high dimensional spaces, having a tendency to lead to highly autocorrelated walker trajectories, and getting stuck in local minima. In cosmology, where the datasets often require us to run inference techniques on high dimensional datasets, the inefficiency of Markov Chain Monte Carlo methods can be severe. Hamiltonian Monte Carlo methods use gradient information about the likelihood as a momentum term for directing the trajectory of the sampler, which results in more efficient sampling. This, however, comes at the cost of computing gradients with respect to the likelihoods. If it computationally intensive to compute these gradients or if the problem requires approximations to be made that are not very accurate, then there is a trade-off between using Hamiltonian Monte Carlo methods and traditional Markov Chain Monte Carlo methods. The ability to compute exact gradients with respect to complex likelihoods in a highly optimized way using differentiable JAX models can enable more optimal usage of gradient-based inference techniques, which can improve and speed up data analysis in cosmology.

Here, I implement the Hamiltonian Monte Carlo No-U-Turn Sampler algorithm using the NumPyro package in Python. This package implements a number of inference techniques that use JAX as a backend and are compatible with the in-built vectorization, parallelization, and just-in-time compilation methods in JAX. I select a uniform priors on Ω_m, h, and w_0. The parameterization of Λ-CDM in JAX-Cosmo does not have Ω_Λ as a free parameter, but I include the derived posterior for Ω_Λ from the other parameters in Figure 3. I use 6 walkers with a burn-in period of 4000 steps and 8000 sampling steps. Figure 3 contain the posteriors we derive from running the HMC algorithm

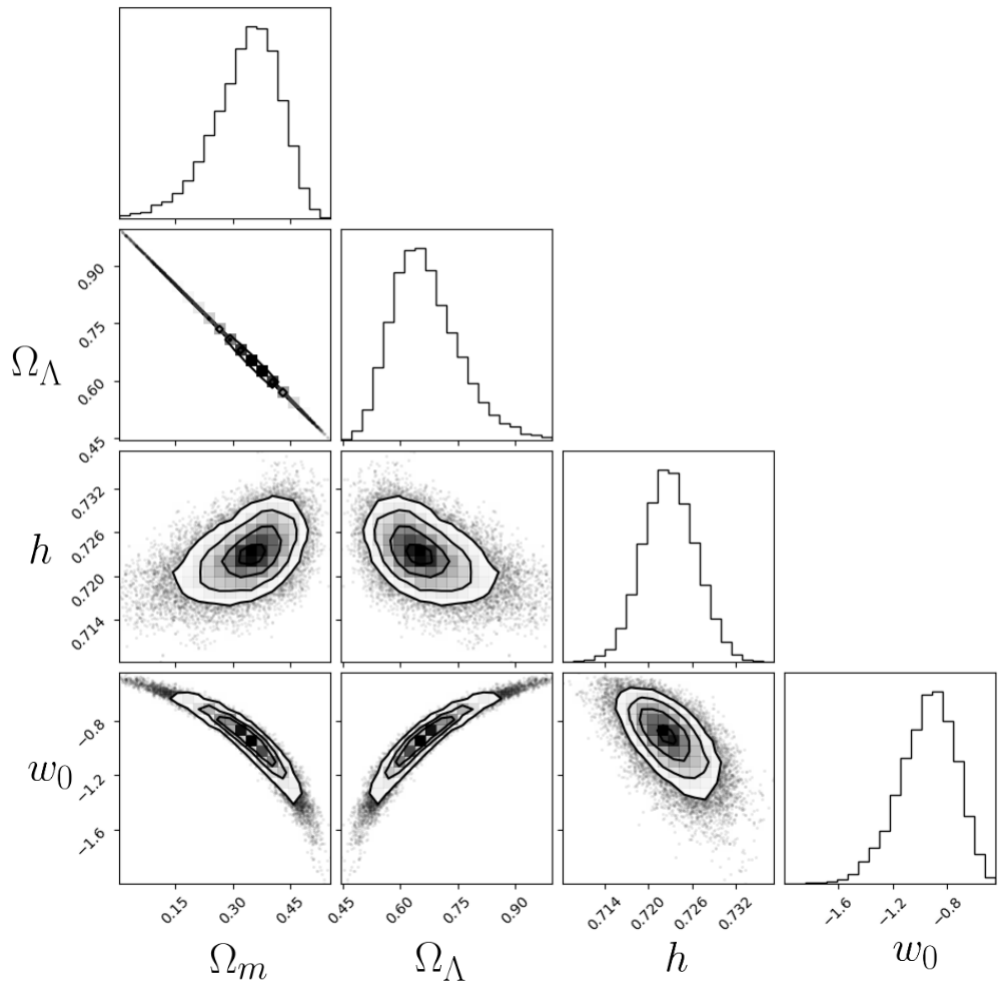

Figure 3: The posterior probability distribution of cosmological parameters Ω_m, Ω_Λ, h, and w_0 using the No U-Turn Sampler Hamiltonian Monte Carlo method on Type 1a Supernovae.

A table summarizing the posteriors from the No-U-Turn Sampler are provided below.

Parameter	Mean	Standard Deviation
Ω_m	0.668	0.088
Ω_Λ	0.332	0.088
H_0	0.723	0.004
w_0	-0.947	0.209

Note that since Ω_Λ was not a free parameter in the parameterization of Λ-CDM in JAX-Cosmo, its standard deviation matches that of Ω_m. The results of the sampler are roughly consistent with the results from the collaboration's fits to the cosmological parameters!

3 Fisher Matrix Comparison

Another important tool in cosmology that requires gradient computations are Fisher forecasts. Via the Cramer-Rao bound, Fisher matrices provide a bound on the constraining power of a probe. In the linked code file, I compute the Fisher forecast for a Type 1a Supernovae probe that observes supernovae that are gaussian distributed in redshift. Based on the Fisher formalism, for this probe, we compute the entries to the Fisher matrix, F as

$$F_{ij} = \sum_{n=1}^{N} \frac{1}{\sigma_{\mu,n}^2} \frac{\partial \mu(z_n)}{\partial p_i} \frac{\partial \mu(z_n)}{\partial p_j}. \tag{6}$$

Figure 4 shows the Fisher forecast for the constraining power of the supernovae probe for cosmological parameters Ω_m and w_0. This is overlaid on top of the posterior we found in Section 2 from the sampling the Type 1a Supernovae dataset. We use $\sigma_\mu = 0.3$. The Pantheon+SH0es average value for $\sigma_\mu \approx 0.25$, so we use a value slightly greater to get a more conservative estimate. In Section 4, we explore a redshift-dependent error model.

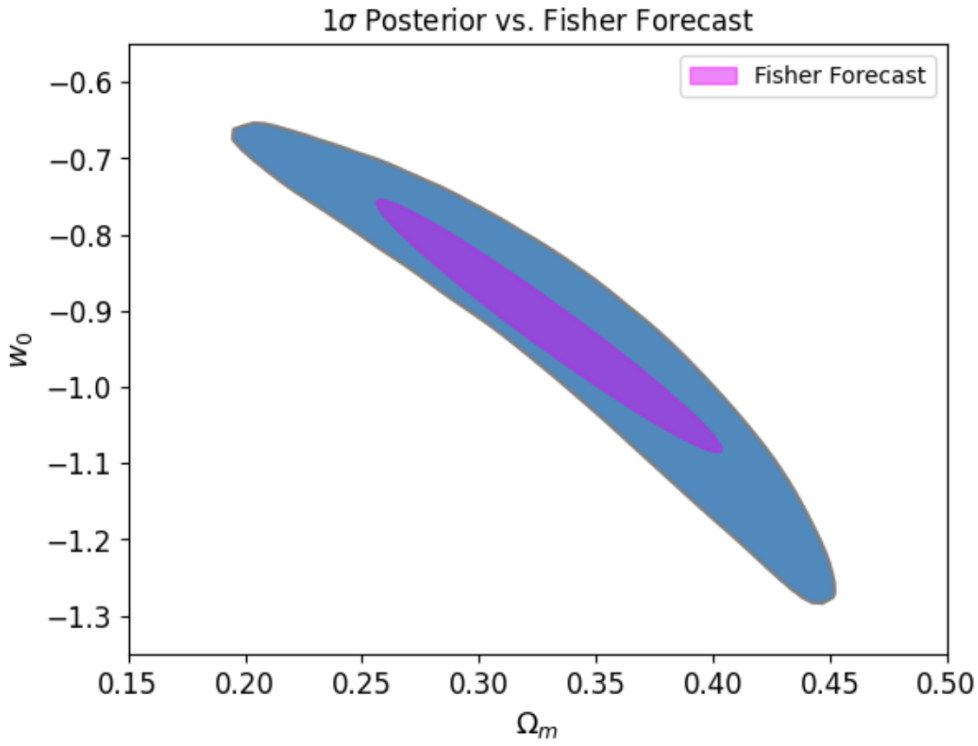

Figure 4: The 1σ posterior distribution for Ω_m vs. w_0 from the Hamiltonian Monte Carlo Sampling in Section 2 is plotted in blue. The Fisher forecast ellipse is overlaid.

We see that the constraints that we forecast with the Fisher analysis are slightly tighter than what we find from sampling the data. Some sources of error here could be modeling the redshift distribution of

supernovae from the probe as a gaussian. The redshifts of the supernovae from the Pantheon+SH0es data are heavily weighted toward low redshifts and falls off toward higher redshifts. Other Fisher forecasts can be found in the linked Python notebooks, but this one is shown as an example.

Fisher analyses are an important use case of JAX and for differentiable cosmology. The efficient methods for autodifferentiation implemented in JAX and ability to calculate gradients with respect to complex models non-analytical models means that it can drastically speed up and increase the complexity of Fisher forecasts. This example of creating a Fisher forecast for a Type 1a Supernovae probe is a relatively simple example. One could imagine creating forecasts for probes of the CMB with a differentiable Boltzmann solver, where the efficiency and accuracy improvements by using JAX will be even more apparent. Even so, in this more simple example, we still see the advantages of using JAX for Fisher forecasts. In the Python notebook, we compare the time speed up for using JAX vs. calculating numerical derivatives with finite differences.

4 Cosmological Probe Optimization

One important use case of differentiable cosmology codes is to formulate and solve optimization problems. Typically, optimization problems involve some form of gradient descent, which become easy to work with in a differentiable cosmology framework. In this section, I demonstrate how we can use JAX to efficiently solve optimization problems. I chose an example that has a loss function that particularly highlights the efficiency speed up that we get from using differentiable models. In this example, I consider optimizing a probe for constraining the dark energy equation of state parameters. Since the are of the Fisher forecast ellipse for two parameters is representative of the constraining power of the probe modeled in the Fisher forecast, the loss function that I aim to minimize is the area of the Fisher ellipse for parameters w_0 and w_a. This technique of minimizing Fisher ellipse areas, and similar one, have also been used in the literature [9]. It also provides a nice proof-of-concept of the usefulness of JAX for solving optimization problems because not only will we be taking gradients of the loss function, but the loss function itself involves the Hessian matrix of the likelihood, which is computed in order to find the area of the Fisher ellipse.

In developing a cosmological probe, it may desirable to optimize the redshift range of the survey, the central redshift of the survey, the aperture size, or any other survey parameters that would affect the cosmological constraints derived from the survey. In this example, we allow the central redshift and the redshift width to vary. We use a redshift-dependent model of errors

$$\Delta m = 0.02\left(\frac{1.7}{z_{\max}}\right)\left(\frac{1+z}{2.7}\right)$$

that was developed for a similar purpose by Linder and Huterer [10]. In optimizing the parameters of the survey to put the best constraints possible on the dark energy equation of state parameters, this

redshift-dependent error creates a trade-off for the optimizer, as moving the central redshift of the survey to higher redshifts or widening the redshift width of the survey leads to higher errors (for a fixed number of Type 1a Supernovae observations). In Figure 5, we show computing the derivative of the loss function (the area of the Fisher ellipse for parameters w_0 and w_a) with respect to the central redshift of the survey for three different survey redshift widths. Since this is a first derivative plot, the redshift at which the first derivative crosses 0 represents the minima of the loss function, i.e. the optimal survey parameter for constraining w_0 and w_a with this model of a supernovae survey.

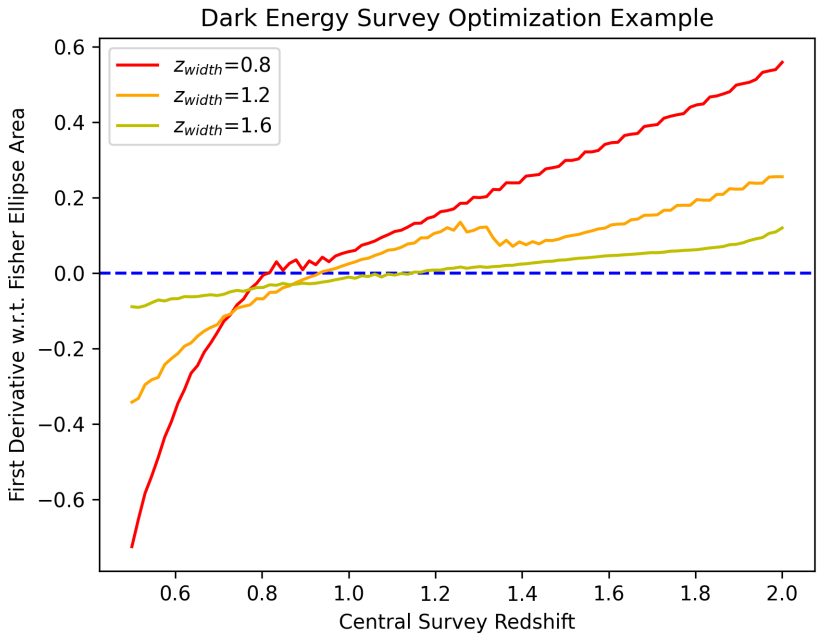

Figure 5: Derivative of the area of the Fisher ellipse for parameters w_0 and w_a with respect to the central survey redshift for different redshift widths for the survey. The redshift of each zero-crossing represents the central redshift of the survey at which we would get the minimum area Fisher ellipse.

Then, we run an optimizer on the loss function, which is quite complex and involves a large number of derivative computations because the loss function itself depends on a Hessian matrix. In Figure 6, we show how the Fisher forecast for the initial guess survey parameters differs from the Fisher forecast for the optimized survey parameters.

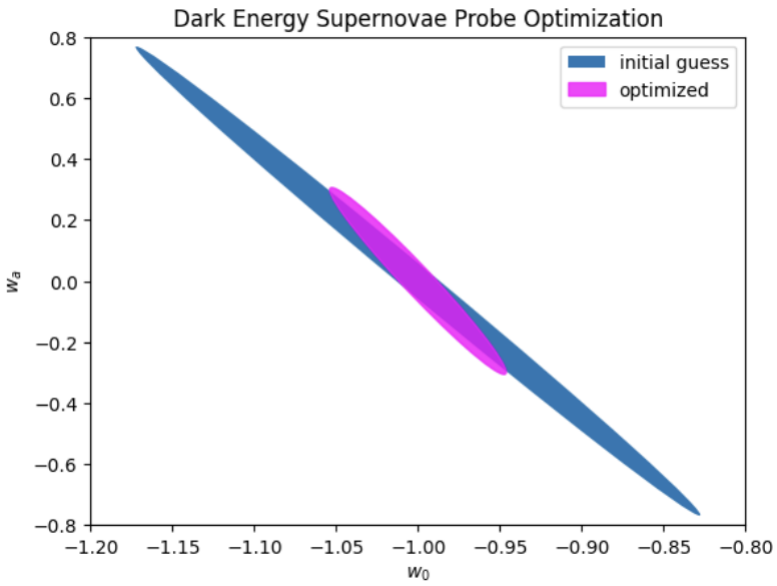

Figure 6: The Fisher forecast for the constraining power of a supernovae probe on parameters w_0 and w_a for initial guess survey parameters vs. the the optimized survey parameters.

Notably, this entire optimization only takes about 6-7 seconds to converge on CPU, which is a testament to how powerful JAX can be in speeding up computations and making computations involving gradients that may have been computationally intractable previously very reasonable to run. To run this sort of optimization using finite differences would have been much slower than using end-to-end automatically differentiable models. Although this represents a drastic idealization of a supernovae probe optimization, the fast convergence given the complexity of the loss function provides strong proof-of-concept for the capabilities of solving complex optimization problems in Cosmology with JAX. With a differentiable Boltzmann solver, one could envision emulating this process for more complicated CMB observables.

5 Differentiating Through Cosmological Simulations

Finally, one of the exciting use cases of differentiable cosmology that was not shown explicitly in the JAX-Cosmo paper is the ability to differentiate through complex models with respect to input parameters. This was already shown in Section 2 for the model of cosmological distance modulus. In this section, we show it for a more complex example of the matter power spectrum. Although JAX-Cosmo is currently not equipped with a differentiable Boltzmann solver, it has a differentiable version of the Eisenstein and Hu transfer function in-built [11]. This enables us to compute the linear and non-linear matter power spectra in very few lines of code (see linked Python notebooks). Figure 7 shows linear matter power spectrum and the derivative of it with respect to Ω_b. We see that matter

power spectrum is most sensitive (highest absolute value of the first derivative) to Ω_b at $k \approx 2 \times 10^{-2}$ h/Mpc, which is the wavenumber corresponding to the size of the horizon at matter-radiation equality, as we would expect. We also see small oscillations in the first derivative at higher values of k, likely corresponding to acoustic effects.

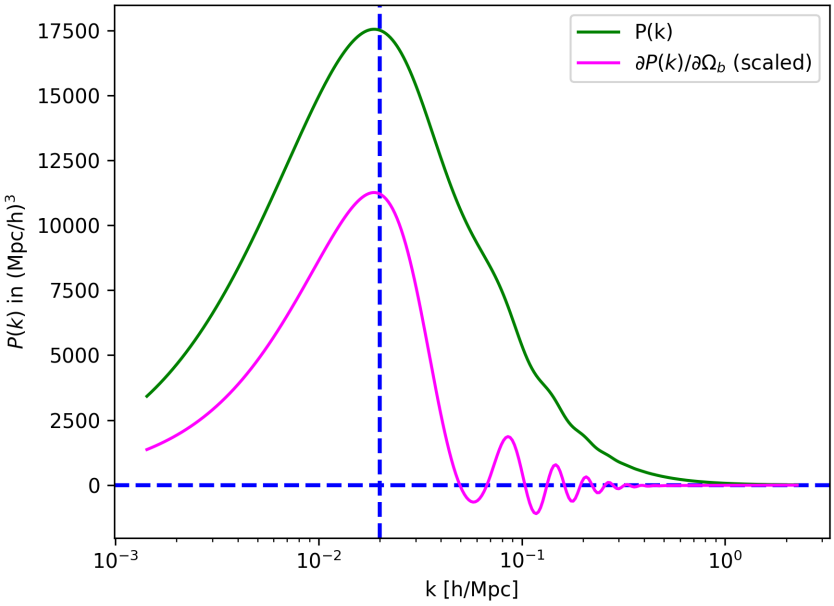

Figure 7: Derivative of the matter power spectrum with respect to Ω_b.

The sensitivity of the matter power spectrum with respect to Ω_b is quite intuitive. I also an example in Figure 8 of computing the derivative of the linear matter power spectrum with respect to σ_8, which describes RMS amplitudes of the matter power spectrum at 8 Mpc/h.

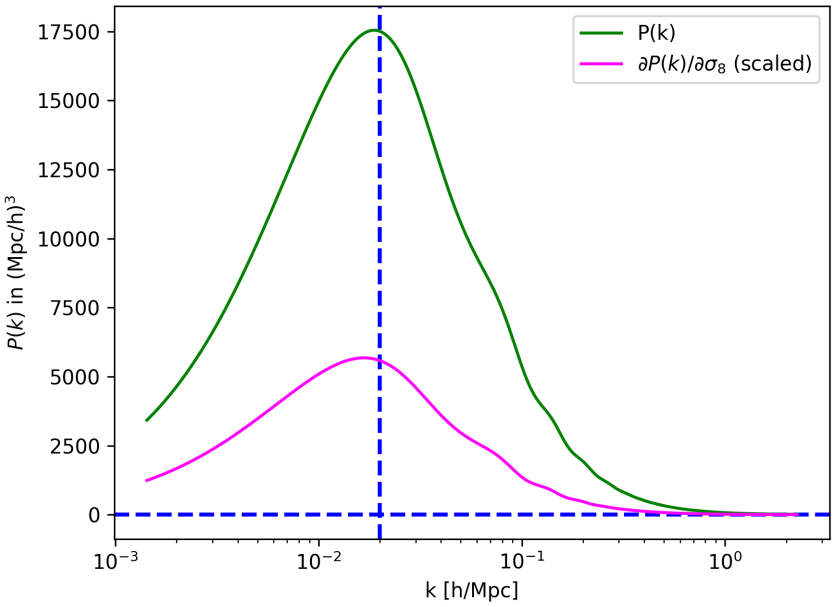

Figure 8: Derivative of the matter power spectrum with respect to σ_8.

This is an exciting tool, which may be useful in more complicated cosmological simulations where parameter dependencies may be less clear than they are here. It was also a great pedagogical tool for me for better understanding the dependencies the matter power spectrum on different parameters.

6 Conclusions

In conclusion, in this project, I have surveyed many examples of important use cases of differentiable cosmology simulations. These included running gradient-based inference techniques to constrain cosmological parameters, calculating Fisher forecasts, optimizing cosmological probes for constraining parameters with a toy example, and differentiating cosmological models with respect to input parameters. Throughout this project, as is clear when you run the code and observe the time speed up from using JAX, the efficiency improvements that we get from using JAX become very apparent. Although I was limited to somewhat simple examples because of the limited methodologies implemented in JAX-Cosmo, I hope that in the future, I can use this library to run much more complicated analyses involving differentiable Boltzmann solvers.

7 Python Notebooks and Slideshow Presentation

Notebook 1:
https://colab.research.google.com/drive/1_yms3i1KU2hi3D1raQ6Ujz50l4yjribF?usp=sharing

Notebook 2:
https://colab.research.google.com/drive/15zyc2k_2UjVC2SxkU_xk7pGWddyJ500l?usp=sharing

Slideshow Presentation Link:
https://docs.google.com/presentation/d/1WQ8YgtUe0yNnrIrbbdOOjpYCkg1PcRDh_1cDFUXIxEM/edit?usp=sharing

8 Acknowledgements

I would like to thank Professor Dvorkin for leading an extremely engaging course in cosmology. I would also like to thank Çağan for all of his help on assignments, in section, and his feedback on my final project.

References

[1] Samuel S. Schoenholz and Ekin D. Cubuk. Jax, m.d.: A framework for differentiable physics, 2020.

[2] Dmitrii Kochkov, Jamie A. Smith, Ayya Alieva, Qing Wang, Michael P. Brenner, and Stephan Hoyer. Machine learning–accelerated computational fluid dynamics. *Proceedings of the National Academy of Sciences*, 118(21), may 2021. doi: 10.1073/pnas.2101784118. URL https://doi.org/10.1073%2Fpnas.2101784118.

[3] James Bradbury, Roy Frostig, Peter Hawkins, Matthew James Johnson, Chris Leary, Dougal Maclaurin, George Necula, Adam Paszke, Jake VanderPlas, Skye Wanderman-Milne, and Qiao Zhang. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

[4] Jean-Eric Campagne, François Lanusse, Joe Zuntz, Alexandre Boucaud, Santiago Casas, Minas Karamanis, David Kirkby, Denise Lanzieri, Yin Li, and Austin Peel. Jax-cosmo: An end-to-end differentiable and gpu accelerated cosmology library, 2023.

[5] Dan Scolnic et al. The pantheon analysis: The full data set and light-curve release. *The*

Astrophysical Journal, 938(2):113, oct 2022. doi: 10.3847/1538-4357/ac8b7a. URL https://doi.org/10.3847%2F1538-4357%2Fac8b7a.

[6] Adam G. Riess et al. A comprehensive measurement of the local value of the hubble constant with 1 km/s/mpc uncertainty from the hubble space telescope and the SH0es team. *The Astrophysical Journal Letters*, 934(1):L7, jul 2022. doi: 10.3847/2041-8213/ac5c5b. URL https://doi.org/10.3847%2F2041-8213%2Fac5c5b.

[7] Eric V. Linder. Exploring the expansion history of the universe. *Phys. Rev. Lett.*, 90:091301, 2003. doi: 10.1103/PhysRevLett.90.091301.

[8] Matthew D. Hoffman and Andrew Gelman. The no-u-turn sampler: Adaptively setting path lengths in hamiltonian monte carlo, 2011.

[9] Z. Haiman. Probing the cosmic dark age in x-rays. In *AIP Conference Proceedings*. AIP, 2001. doi: 10.1063/1.1434627. URL https://doi.org/10.1063%2F1.1434627.

[10] Eric V. Linder and Dragan Huterer. Importance of supernovae at z ¿ 1.5 to probe dark energy. *Physical review. D, Particles and fields*, 67(8), 2003. ISSN 0556-2821.

[11] Daniel J. Eisenstein and Wayne Hu. Baryonic features in the matter transfer function. *The Astrophysical Journal*, 496(2):605–614, apr 1998. doi: 10.1086/305424. URL https://doi.org/10.1086%2F305424.