5-4-2021

Cosmological Inflation in N-Dimensional Gaussian Random Fields with Algorithmic Data Compression

Connor A. Painter
University of Richmond

COSMOLOGICAL INFLATION IN N-DIMENSIONAL GAUSSIAN RANDOM FIELDS WITH ALGORITHMIC DATA COMPRESSION

BY

Connor A. Painter

An Honors Thesis

Submitted to:

Department of Physics
University of Richmond
Richmond, VA

May 4, 2021

Advisor: Dr. Ted Bunn

Cosmological Inflation in N-Dimensional Gaussian Random Fields with Algorithmic Data Compression

Connor A. Painter
Physics Department, University of Richmond, Richmond, VA 23173, USA

The leading modern theories of cosmological inflation are increasingly multi-dimensional. The "inflaton field" $\boldsymbol{\phi}$ that has been postulated to drive accelerating expansion in the very early universe has a corresponding potential function V, the details of which, such as the number of dimensions and shape, have yet to be specified. We consider a natural hypothesis that V ought to be maximally random. We realize this idea by defining the V as a Gaussian random field in some number N of dimensions. We repeatedly simulate of the evolution of $\boldsymbol{\phi}$ given a set of conditions on the "landscape" of V. We simulate a "path" stepwise through $\boldsymbol{\phi}$-space while simultaneously computing V and its derivatives along the path via a constrained Gaussian random process, incorporating the information from prior steps. When N is large, this method significantly reduces computational load as compared to methods which generate the potential landscape all at once. Even so, computation of the covariance matrix $\boldsymbol{\Gamma}$ of constraints on V can quickly become intractable. Inspired by this problem, we present data compression algorithms to prioritize the necessary information already simulated, then keep an arbitrarily large portion. Information such as the evolution of the scale factor and tensor and scalar perturbations can be extracted from any particular path, then statistical information about these quantities can be gathered from repeated trials. In these ways, we present a versatile multi-variable program for exploration into how accurately this emergent model can fit to observation.

I. INTRODUCTION

Characterization of plausible inflationary models is an extremely high priority in cosmology. Since inflationary theory was developed in the late 1970s and 80s, the cosmological community has sought to constrain its main free parameter: the inflaton potential. There now exist *libraries* of invented potentials which have been exhaustively simulated with various initial conditions in efforts to gauge how accurately each of their predictions align with modern cosmological data. Some model potentials have been shown to be ineffective representatives of our physical universe, while a surprising (and concerning, to some) number persistently conjure predictions that fall within present uncertainties. Computational intensity is a critical factor behind which models have been tested most thoroughly. Potentials with high numbers of degrees of freedom N have often been omitted from analysis or reduced to lower dimensions in favor of more detailed solutions to differential equations. The N-dimensional Gaussian random potential is an especially bulky choice since the simulation of any new potential value must incorporate all other information known about the potential in the form of complicated linear systems. In the following sections, I motivate and present methods for exploration into the Gaussian random potential model, including data compression and simplification specifically adapted for this problem, as well as preliminary findings from our own supercomputer simulations. I conclude with the status of the project and a short list of ideas for future exploration. We begin with a review of some relevant cosmology.

II. INFLATIONARY THEORY AND NOTATION

Most of the content of this project is accessible through the premises of basic inflationary theory. We assume the reader has some elementary knowledge of multivariate calculus, linear algebra, probability, and differential equations. However, we will not assume any prior knowledge of inflation. We proceed by reviewing a motivation behind the theory of inflation, examining an inconsistency between the Big Bang model of expansion and observations in microwave astronomy.

A. The Horizon Problem

The Cosmic Microwave Background (CMB), accidentally discovered in the 1960s, is faint microwave radiation detectable at every unobstructed point in the sky [2]. It is a remnant of one of the early stages in the evolution of the universe, the epoch of recombination, and it offers important insights in early-universe cosmology. The CMB is very nearly uniform and may be approximated to a black-body spectrum at a characteristic temperature of 2.72548 ± 0.00057 K. The temperature fluctuates from the mean at various points in the sky ever-so-slightly, on the order of a few parts in ten thousand. The large scale homogeneity of the CMB—the fact that the temperature at one point in the sky deviates so little from the temperature at the polar opposite location—indicates that the entire observable background must have been in contact long enough to reach thermal equilibrium.

The Big Bang theory provides insight on the angular separation θ over which regions of the CMB should be

causally connected. Two regions of space are causally connected if light travelling at speed c has had time to propagate from region A to region B given the expansion of the universe. Two regions that are not causally connected have not exchanged any information, particularly thermal energy, throughout the duration of the universe. For theory to agree with observation, it must predict that the entire CMB is causally connected, i.e. $\theta = 180°$. To verify the Big Bang theory, let's introduce some of its key quantities and measures.

The relative size of our expanding universe at any point in time is conventionally measured by the *scale factor* $a(t)$. The scale factor is defined so that the proper distance d between two objects at two arbitrary times t_1, t_2 is related by

$$\frac{d(t_1)}{d(t_2)} = \frac{a(t_1)}{a(t_2)}. \tag{1}$$

The scale factor is widely accepted to be a monotonically increasing function of time from the Big Bang, $a(t = 0) = 0$, to now, $a(t = t_0) = 1$. In other words, the universe has outwardly expanded continuously since the Big Bang. Specifically, the theory suggests that $a(t)$ evolved in time as follows.

$$a(t) \propto \begin{cases} t^{1/2} & 0 < t \leq 47\ 000 \text{ years} \\ t^{2/3} & 47\ 000 < t \leq 9.8 \text{ billion years} \\ \exp(Ht) & 9.8 \text{ billion years} < t \end{cases} \tag{2}$$

The piecewise formulation of a corresponds to eras of different dominant energy densities: radiation, matter, and dark energy, respectively.

Proper distance $d(t)$ between two distant regions of space changes while the universe expands. It is a distance that is hypothetically measured by freezing a moment in time and extending a ruler from one region to the other. In early times, closer to the Big Bang, the ruler might measure two regions to be close together, while today, the ruler would measure a much greater distance. Proper distance is defined in terms of the *comoving distance* χ, part of a convenient system of comoving coordinates that yield distances constant in time by expanding synchronously with the universe. To define these distances rigorously, imagine that we, on Earth, observe photons emitted at t_e from a distant object. The comoving distance between the Earth and that object is

$$\chi = c \int_{t_e}^{t_0} \frac{dt}{a(t)} \tag{3}$$

Billions of years ago, χ was still be the same value we computed today, since comoving coordinates factor out the expansion of the universe. However, the proper distance to the object factors expansion back in as $d(t) = a(t)\chi$. At the time in the past when $a = 0.5$, correspondingly $d = 0.5\chi$.

Another intimately related quantity is the horizon distance $d_{\text{hor}}(t)$, which measures the diameter of the causally connected universe at any time t. If the proper distance between two objects at time t is larger than $d_{\text{hor}}(t)$, then those objects were never causally connected. The horizon distance can be expressed as an integral similar to (3):

$$d_{\text{hor}}(t) = c \int_0^t \frac{dt}{a(t)}$$

The horizon distance today is equal to the comoving distance to a photon emitted exactly at $t = 0$ which has traveled unhindered since then:

$$d_{\text{hor}}(t_0) = c \int_0^{t_0} \frac{dt}{a(t)} = 46.9 \text{ billion light years},$$

with $a(t)$ defined as in the traditional Big Bang theory equation (2).

Finally, we are prepared to verify observations that the CMB is causally connected. The CMB radiation was emitted at the "surface of last scattering", $t = t_{\text{ls}} \approx$ 379 000 years. The angular separation between causally connected portions of the CMB is calculable approximately in terms of $d_{\text{hor}}(t_{\text{ls}})$ and one other important quantity, the cosmological redshift $z(t)$:

$$\theta \approx \frac{d_{\text{hor}}(t_{\text{ls}})}{d_{\text{hor}}(t_0)} z(t_{\text{ls}}). \tag{4}$$

Cosmological redshift is the factor by which the wavelength of light has stretched out due to the expansion of the universe. The approximation (4) is only valid because $z(t_{\text{ls}}) = 1100$, which is much greater than 1. Once again, the Big Bang model can be used to produce a value for the horizon distance at the surface of last scattering:

$$d_{\text{hor}}(t_{\text{ls}}) = c \int_0^{t_{\text{ls}}} \frac{dt}{a(t)} \approx 1.31 \text{ million light years}.$$

Substituting, we find $\theta \approx 2°$, woefully short of the required 180° angular separation to explain observations [8].

This failure of the Big Bang theory can be blamed on the scale factor, which completely determines how quickly each point on the last scattering surface has spread away from the others. A clever individual might propose that equation (2) is wrong, somehow, and that the universe underwent more rapid growth in early times than originally theorized. In fact, that turns out to be the proposed solution. The horizon problem, along with other similarly puzzling problems left behind by the Big Bang model, can be explained by assuming there existed a period of rapid expansion in the very early universe, called *inflation*. Prominent inflationary theories suppose that the scale factor must have increased by a factor of 10^{22} to 10^{26} between $t = 10^{-36}$ and 10^{-32} seconds, an

incredible leap. The details of the mechanism behind inflation are still unknown, but most models begin by proposing a potential function, which dominates the energy density of the universe at early times, and an associated scalar field evolution (more in part II B). In any case, such a dramatic expansion in the early universe would explain the horizon problem: before inflation, regions of space which now encompass the entire surface of last scattering settled into thermal equilibrium, then inflation blasted those regions far away from one another at rates we failed to conjecture, at first.

Now that inflation is properly motivated, we can examine more closely the assumption of an inflation-driving potential function. The specifics of an assumption turn out to be very important in the subsequent evolution of its associated model universe.

B. The Trajectory

Suppose there existed some "inflaton" field $\boldsymbol{\phi}(t)$ which permeated the early universe homogeneously in space, but whose time evolution was governed by its associated potential function $V(\boldsymbol{\phi})$. In general, $\boldsymbol{\phi} = (\phi^{(0)}, \phi^{(1)}, ..., \phi^{(N-1)})$ is a vector of N components while V is scalar-valued. In addition to its existence, we assert that the inflaton potential dominates the energy density of the early universe so that its characteristics may determine the particularities of large-scale cosmic expansion in early times. Derived from the Friedmann Equation, the Fluid Equation, and equations in field theory, the differential equation for the evolution of $\boldsymbol{\phi}$ in terms of V is

$$\ddot{\boldsymbol{\phi}} + 3H\dot{\boldsymbol{\phi}} + \nabla_\phi V = 0 \tag{5}$$

where overdots signify derivatives with respect to cosmic time t, $H = \dot{a}/a$ is the Hubble parameter, and Planck units $c = \hbar = 8\pi G = 1$ are implemented throughout. Component-wise, (1) reads

$$\ddot{\phi}^{(\alpha)} + 3H\dot{\phi}^{(\alpha)} + \frac{\partial V}{\partial \phi^{(\alpha)}} = 0 \tag{6}$$

where $\alpha = 0, 1, ..., N-1$. This equation recalls the familiar intuition of a ball rolling around on a surface with friction, an analogy we can carry with us through this paper. If $\boldsymbol{\phi}$ is the position of the ball in N-dimensional space (for $N = 2$, imagine that $(\phi^{(0)}, \phi^{(1)}, V)$ is analogous to (x, y, z)), then the ball accelerates *downhill* in response to slopes $\nabla_\phi V$, but is resisted by a frictional term $3H\dot{\boldsymbol{\phi}}$.

In practice, it is often useful to measure time in terms of the expansion of the universe [6]. During inflation, it is often the case that $a(t) \approx A \cdot \mathrm{Exp}(Ht)$ with $A, H > 0$, so a natural unit of time is the "number of e-folds"

$$N_e(t) = \ln\left(\frac{a(t)}{a(t_i)}\right) \tag{7}$$

where t_i is the arbitrary start of inflation. Some algebra transforms (7) into

$$\frac{d^2\phi^{(\alpha)}}{dN_e^2} + (3-\epsilon)\frac{d\phi^{(\alpha)}}{dN_e} + \frac{1}{H^2}\frac{\partial V}{\partial \phi^{(\alpha)}} = 0 \tag{8}$$

noting that $H = \dot{a}/a$. To cleverly track the progression of inflation in our models, we have defined the "slow-roll" parameter ϵ as

$$\epsilon \equiv -\frac{\dot{H}}{H^2} = \frac{1}{2}\left|\frac{d\boldsymbol{\phi}}{dN_e}\right|^2 \tag{9}$$

which signals the end of inflation when $\epsilon \uparrow 1$.

Generally, V is specified by the theorist (more in III.), who then solves (8) with various initial conditions $\boldsymbol{\phi}_0, d\boldsymbol{\phi}_0/dN_e$ for a path evolution $\boldsymbol{\phi}(t)$ through critical periods of inflation. For V to represent a viable inflationary model, there are some general requirements by which it must abide. Vaguely, models must predict that the interaction of $\boldsymbol{\phi}$ in V initiates a period of rapid expansion, then diminishes in energy until other densities of energy (radiation, matter, dark) begin to dominate the changes in $a(t)$. The relationship between $a(t)$, $\boldsymbol{\phi}(t)$, and $V(t)$ as presented by the Friedmann equation can guide us toward ensuring this evolution plays out.

$$3H(t)^2 = 3\frac{\dot{a}(t)}{a(t)} = \frac{1}{2}|\dot{\boldsymbol{\phi}}(t)|^2 + V(t) \tag{10}$$

This differential relationship motivates the following conditions on V:

1. *The slow-roll condition.* The potential must contain some gradual and gradually changing slopes $\partial V/\partial\phi^{(\alpha)}$ for a sufficient length along the evolution of $\boldsymbol{\phi}$. These assumptions intend to allow $V \approx$ constant and $\dot{\boldsymbol{\phi}} \approx$ constant as terminal velocity is reached along a relevant duration. As a result, (10) reads that $\dot{a}/a \approx$ constant and the simplified differential equation solves to $a \propto \mathrm{Exp}(Ht)$.

2. *The minimum condition.* The potential must have a local minimum at $V = 0$ in which the $\boldsymbol{\phi}$ solution settles. If $\boldsymbol{\phi}$ settles to some $V_{min} \neq 0$ after its required inflationary stretch, a will either continue to increase exponentially ($V > 0$, eternal inflation) or start to decrease exponentially ($V < 0$, universe collapses). This can be seen by referring, again, to (10) and noting that $\dot{\boldsymbol{\phi}} \approx 0$ when settling in a minimum. The end behavior of these models will always be $\dot{a}/a \approx V_{min}$, so any nonzero minimum would not lead to present observations.

As long as the choice of V satisfies these requirements, it is worthwhile to test further.

C. An Example: The Quadratic Model

Perhaps the simplest model potential that yields reasonable solutions is the N-dimensional quadratic potential

$$V(\boldsymbol{\phi}) = \frac{1}{2}\sum_{\alpha=0}^{N-1}\left(k_\alpha \phi^{(\alpha)}\right)^2$$

for some constants k_α. Notice that with low enough k_α, the slow-roll condition is satisfied and the minimum condition is always satisfied since $V(\mathbf{0}) = 0$. For illustrative purposes, we can briefly solve for the trajectory of $\boldsymbol{\phi}$ evolving under this simple potential. We seek to solve (8), which now reads

$$\frac{d^2\phi^{(\alpha)}}{dN_e^2} + (3-\epsilon)\frac{d\phi^{(\alpha)}}{dN_e} + \frac{k_\alpha^2}{H^2}\phi^{(\alpha)} = 0 \tag{11}$$

with arbitrary initial conditions $\phi^{(\alpha)}(0) = 10$ and $\dot{\boldsymbol{\phi}}(0) = \mathbf{0}$. Like in most relevant inflationary models, this differential equation is not solvable analytically, so we turn to numerical methods. We implement a simple construction of this quadratic model in Python (we use this programming language throughout the project), particularly using the `solve_ivp` function within the `scipy.integrate` class. This function integrates the equations of motion (8) from $t = 0$ until a stopping condition is met (more in IV) and yields discrete points in $\boldsymbol{\phi}$, $d\boldsymbol{\phi}/dN_e$, and V. A compilation of trajectory plots for certain k_α and N are found in Figure 1.

D. Observable Quantities

If an inflationary model satisfies the elementary conditions, its predictions can be evaluated against observations [5]. Specifically, we can extract the following relevant quantities for comparison with modern consensus:

- The total amount of inflation. For an inflationary model to solve problems left behind by the traditional Big Bang Theory (such as the Horizon Problem), the universe has to expand enough during the relevant period. It is generally accepted that 60 e-folds of inflation must have occurred. That is $N_e(t_f) \geq 60$.
- Scalar (matter) perturbations. In the early universe, quantum fluctuations are theorized to have distorted space in ways that are observable today [3]. Each inflationary model evolves these fluctuations through time in unique ways; different inflationary models yield different predictions about their present-day appearance. We can briefly sketch out the connection between inflation and these primordial perturbations:

The metric tensor $\boldsymbol{g}(\boldsymbol{x})$ describes the curvature of space at every point in space.

$$\boldsymbol{g}(\boldsymbol{x}) = \begin{bmatrix} g^{00}(\boldsymbol{x}) & g^{01}(\boldsymbol{x}) & g^{02}(\boldsymbol{x}) \\ g^{10}(\boldsymbol{x}) & g^{11}(\boldsymbol{x}) & g^{12}(\boldsymbol{x}) \\ g^{20}(\boldsymbol{x}) & g^{21}(\boldsymbol{x}) & g^{22}(\boldsymbol{x}) \end{bmatrix} \tag{12}$$

The superscripts span over coordinates in three-dimensional space, e.g. $(0,1,2) = (r,\theta,\phi)$, so $\boldsymbol{g}$ is fully capable of describing curvature along each of the coordinate axes. In general, the metric tensor describes the curvature of space*time*, but we are confining our attention to the spatial indices for now. One of the most important metric tensors is the flat space metric $\boldsymbol{\eta}$, which is just the identity.

$$\boldsymbol{\eta} = \mathbb{I}_3 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{13}$$

With appropriate choices for coordinates, a metric tensor can be decomposed into a flat term and its perturbations around flatness.

$$\boldsymbol{g} = \boldsymbol{\eta} + \delta\boldsymbol{g} \tag{14}$$

The perturbations can be diagonalized as

$$\delta\boldsymbol{g} = \boldsymbol{P}\begin{bmatrix} \lambda_0 & 0 & 0 \\ 0 & \lambda_1 & 0 \\ 0 & 0 & \lambda_2 \end{bmatrix}\boldsymbol{P}^{-1} = \boldsymbol{P}\boldsymbol{\Lambda}_{\delta\boldsymbol{g}}\boldsymbol{P}^{-1} \tag{15}$$

where λ_i are eigenvalues of $\delta\boldsymbol{g}$. The diagonalization $\boldsymbol{\Lambda}_{\delta\boldsymbol{g}}$ contains all the relevant information about the deviation in the curvature of space about flatness. We can further decompose $\boldsymbol{\Lambda}_{\delta\boldsymbol{g}}$ into two critically important curvatures: *expansion* and *shearing*.

$$\boldsymbol{\Lambda}_{\delta\boldsymbol{g}} = \underbrace{\frac{1}{3}\mathrm{Tr}(\boldsymbol{\Lambda}_{\delta\boldsymbol{g}})\,\boldsymbol{\eta}}_{\text{Expansion}} + \underbrace{\left(\boldsymbol{\Lambda}_{\delta\boldsymbol{g}} - \frac{1}{3}\mathrm{Tr}(\boldsymbol{\Lambda}_{\delta\boldsymbol{g}})\boldsymbol{\eta}\right)}_{\text{Shearing}} \tag{16}$$

Expansion is curvature of equal magnitude is all directions. It is a scalar quantity still defined at every point in space, i.e. $\mathrm{Tr}(\boldsymbol{\Lambda}_{\delta\boldsymbol{g}}) = \mathrm{Tr}(\boldsymbol{\Lambda}_{\delta\boldsymbol{g}})(\boldsymbol{x})$. Expansion (or contraction) of space is largely caused by matter, thus perturbations that expand and contract space are sometimes called matter perturbations. Most of the time, these scalar perturbations are not expressed as a function of position, but as a function of wavevector $\boldsymbol{k}$.

$$\mathcal{P}_S(\boldsymbol{k}) = \left|\texttt{FourierTransform}\left[\mathrm{Tr}(\boldsymbol{\Lambda}_{\delta\boldsymbol{g}})\right]\right|^2 \tag{17}$$

This is the power spectrum of scalar perturbations. The perturbations are said to be "scale-invariant" since they generally look the same at every time during inflation. Mathematically, this means the power spectrum follows the power law

$$\mathcal{P}_S(\boldsymbol{k}) = A_S\left(\frac{|\boldsymbol{k}|}{k_0}\right)^{n_S-1} = A_S\left(\frac{k}{k_0}\right)^{n_S-1} \tag{18}$$

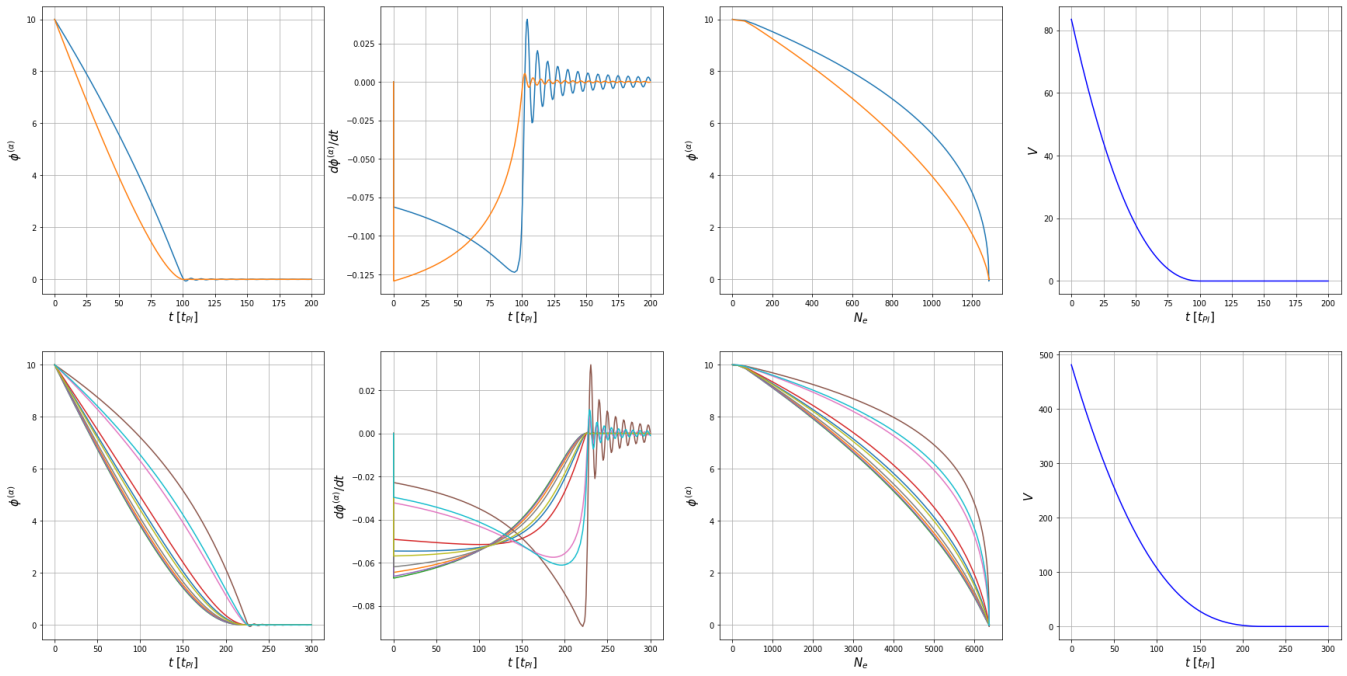

FIG. 1. Sample trajectories from the N-dimensional quadratic potential with $k_\alpha \sim \mathcal{N}(1, 0.3)$. The first row is an example with $N = 2$ while the second row sets $N = 10$. The columns plot, from left to right, $\boldsymbol{\phi}(t)$, $\dot{\boldsymbol{\phi}}(t)$, $\boldsymbol{\phi}(N_e)$, and $V(t)$. Notice that the initial conditions $\phi^{(\alpha)} = 10$ and $\dot{\phi}^{(\alpha)} = 0$ are represented appropriately. Visible on the "velocity" plots is the quick ascent to slow roll of $\dot{\boldsymbol{\phi}}$, relevant inflationary period, then settling to the quadratic minimum.

where A_S is the scalar amplitude, n_S is the scalar spectral index, and $k_0 = 0.05\ \text{Mpc}^{-1}$ is an arbitrary "pivot scale". The math and physics required to compute $\mathcal{P}_S(\boldsymbol{k})$ from the trajectory is quite intense, so we have relocated it in Appendix A for the curious reader. The quantities A_S, n_S have been well constrained in recent years by the Planck missions [1], offering a powerful test of inflationary models.

$$\ln(10^{10} A_S) = 3.040 \pm 0.016 \tag{19}$$

$$n_S = 0.9626 \pm 0.0057 \tag{20}$$

- Tensor perturbations. The shearing term in the curvature decomposition (16) represents how much three-dimensional space is shifted from flat, rectangularly gridded space to grid lines that form parallelepipeds. When space is contorted in this way, the curvature radiates away as gravitational waves. Theoretically, signatures of these gravitational waves should be visible in the CMB with high enough resolution. However, they have yet to be detected. This leads to a constraint in the form of an upper bound on a spectral amplitude. To be specific, we define

$$\boldsymbol{T}_{\delta \boldsymbol{g}}(\boldsymbol{k}) = \texttt{FourierTransform}\left[\boldsymbol{S}_{\delta \boldsymbol{g}}\right] \tag{21}$$

where $\boldsymbol{S}_{\delta \boldsymbol{g}} \equiv \boldsymbol{\Lambda}_{\delta \boldsymbol{g}} - \text{Tr}(\boldsymbol{\Lambda}_{\delta \boldsymbol{g}})/3$. Now, the power spectrum of tensor perturbations $\mathcal{P}_T(\boldsymbol{k})$ is related to the absolute square of the eigenvalues of $\boldsymbol{T}_{\delta \boldsymbol{g}}$. As with scalar perturbations, this power spectrum of tensor perturbations should resemble a power law.

$$\mathcal{P}_T(\boldsymbol{k}) = A_T \left(\frac{|\boldsymbol{k}|}{k_0}\right)^{n_T} = A_T \left(\frac{k}{k_0}\right)^{n_T} \tag{22}$$

Here, A_T is the tensor amplitude and n_T is the tensor spectral index. The constraints from Planck data come in the form of a tensor-to-scalar ratio:

$$r(\boldsymbol{k}) = \frac{\mathcal{P}_T(\boldsymbol{k})}{\mathcal{P}_S(\boldsymbol{k})} = \frac{A_T}{A_S} \cdot \left(\frac{k}{k_0}\right)^{n_T - n_S + 1} \tag{23}$$

Specifically, the tensor-to-scalar ratio is constrained at a particular wavelength $k_t = 0.002\ \text{Mpc}^{-1}$ to roughly

$$r(k_t) < 0.1. \tag{24}$$

In terms of the trajectory, the power spectrum of tensor perturbations is surprisingly easy to calculate.

$$\mathcal{P}_T(\boldsymbol{k}) = \frac{2}{\pi^2} H(k)^2 \tag{25}$$

For an inflationary model, $k(t) \propto a(t)H(t)$ is another monotonically increasing function of time and is, thus, a valid temporal variable at which to measure time-dependent quantities such as H.

The wavenumber represents the "horizon-crossing mode" at time t, or the shearing mode that is comparable in wavelength to the horizon distance at that time during inflation.

These observable quantities will serve as our guides in determining which parameters for our inflationary model, the Gaussian random potential, tend to yield plausible trajectories.

III. THE GAUSSIAN RANDOM POTENTIAL

Our choice of potential is interesting in that it does not have a formulaic definition, but a statistical one. We posit that $V = V(\boldsymbol{\phi}_0, \boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_{N-1})$ is realized via a Gaussian random process with mean zero, variance V_0^2, and coherence scale s [9].

A. Generation: Two Points

Suppose for the moment that $V(\boldsymbol{\phi})$ is a function of just one variable, i.e. $\boldsymbol{\phi} = (\phi^{(0)})$. Now, I want to know the value of a brand-new potential function at just one point $\boldsymbol{\phi}_0$. With no constraints from previous data, the first value is just a random number generated from a Gaussian distribution. We say $V(\boldsymbol{\phi}_0) \sim \mathcal{N}(0, V_0^2)$, read "$V(\boldsymbol{\phi}_0)$ is drawn from a Normal distribution of mean zero and variance V_0^2."

Now that we know the first point, we ask for the value of the potential at another location $\boldsymbol{\phi}_1$. Importantly, $V(\boldsymbol{\phi}_1)$ will not be completely random, but constrained to some extent by $V(\boldsymbol{\phi}_0)$. This constraint comes so that the curves of V are smooth; if every generated point were completely random, V would end up awfully discontinuous and "staticky." The mathematical constraint comes in the form of a *correlation function*,

$$\langle V(\boldsymbol{\phi}_0)V(\boldsymbol{\phi}_1)\rangle = V_0^2 \exp\left(-\frac{|\boldsymbol{\phi}_0 - \boldsymbol{\phi}_1|^2}{2s^2}\right), \tag{26}$$

where angular brackets represent expected value. When $\boldsymbol{\phi}_1$ is very near $\boldsymbol{\phi}_0$, $V(\boldsymbol{\phi}_1)$ is highly constrained to be near $V(\boldsymbol{\phi}_0)$, or else V would have to change unreasonably rapidly. On the other hand, if $\boldsymbol{\phi}_1$ is relatively far from $\boldsymbol{\phi}_0$, then $V(\phi_1)$ "doesn't care much" about $V(\boldsymbol{\phi}_0)$ and should be left mostly to randomness. Notice that (26) includes both the *coherence scale* s and the *inflationary energy scale* V_0. The coherence scale how far is "far" when measuring distance between $\boldsymbol{\phi}_0$ and $\boldsymbol{\phi}_1$ in $\boldsymbol{\phi}$-space. If $|\boldsymbol{\phi}_0 - \boldsymbol{\phi}_1| < s$, then I expect $V(\boldsymbol{\phi}_0) - V(\boldsymbol{\phi}_1)$ to be fairly small; the points are highly correlated. If $|\boldsymbol{\phi}_0 - \boldsymbol{\phi}_1| \gg s$, I have no way of predicting any relationship between $V(\boldsymbol{\phi}_0)$ and $V(\boldsymbol{\phi}_1)$; the points are weakly correlated. The inflationary energy scale is the standard deviation of V if randomly regenerated over and over at the same $\boldsymbol{\phi}$. To see this, notice that (26) reduces to V_0^2 when $\boldsymbol{\phi}_0 = \boldsymbol{\phi}_1$. In a sense to be clarified soon, s^2 and V_0^2 are variances in the $\boldsymbol{\phi}$ and V directions.

To generate $V(\boldsymbol{\phi}_1)$, we construct the *covariance matrix* $\boldsymbol{\Gamma}$ of constraints on V as

$$\boldsymbol{\Gamma} = [\Gamma^{(ij)}] = [\langle V(\boldsymbol{\phi}_i)V(\boldsymbol{\phi}_j)\rangle] \tag{27}$$

where i, j span through all points in $\boldsymbol{\phi}$. So far, $\boldsymbol{\Gamma}$ is 2×2, and $V(\boldsymbol{\phi}_1)$ can be calculated using its components. $V(\boldsymbol{\phi}_1)$ can be considered the sum of two predictions made by $\boldsymbol{\Gamma}$. We can say $V(\boldsymbol{\phi}_1) = \mu + \sigma$, where μ is the expected value of $V(\boldsymbol{\phi}_1)$ and σ is a random deviation from that mean, constrained appropriately. The predicted mean is calculated from $\boldsymbol{\Gamma}$ as

$$\begin{aligned}\mu &= \frac{\Gamma^{(10)}}{\Gamma^{(00)}} V(\boldsymbol{\phi}_0) \\ &= V(\boldsymbol{\phi}_0) \cdot \exp\left(-\frac{|\boldsymbol{\phi}_0 - \boldsymbol{\phi}_1|^2}{2s^2}\right).\end{aligned}$$

The random deviation is slightly more complicated. Define the conditional covariance Γ_C as

$$\begin{aligned}\Gamma_C &= \Gamma^{(11)} - \frac{\Gamma^{(10)}\Gamma^{(01)}}{\Gamma^{(00)}} \\ &= V_0^2 \left(1 - \exp\left(-\frac{|\boldsymbol{\phi}_0 - \boldsymbol{\phi}_1|^2}{s^2}\right)\right)\end{aligned}$$

This quantifies how constrained $V(\boldsymbol{\phi}_1)$ is from $V(\boldsymbol{\phi}_0)$. Notice that if $\boldsymbol{\phi}_1$ and $\boldsymbol{\phi}_0$ are very close together, $\Gamma_C \approx 0$, meaning that there will be almost no deviation from μ. Completing the derivation, sample one number y from a standard normal distribution ($y \sim \mathcal{N}(0, 1)$), then set

$$\sigma = y\sqrt{\Gamma_C}$$

and $V(\boldsymbol{\phi}_1) = \mu + \sigma$.

For example, let's say I sample $V(\boldsymbol{\phi}_0)$ at $f_1 V_0$ and I want to sample V again at $\boldsymbol{\phi}_1 = \phi_0^{(0)} + f_2 s$, for some fractions f_1, f_2. In this case, I have

$$\begin{aligned}\mu &= f_1 V_0 \exp\left(-\frac{f_2^2}{2}\right), \\ \Gamma_C &= V_0^2 \left(1 - \exp(-f_2^2)\right), \text{and} \\ \sigma &= y \cdot V_0 \sqrt{1 - \exp(-f_2^2)},\end{aligned}$$

which evaluate $V(\boldsymbol{\phi}_1)$ to

$$V(\boldsymbol{\phi}_1) = V_0 \left(f_1 \exp\left(-\frac{f_2^2}{2}\right) + y\sqrt{1 - \exp(-f_2^2)}\right).$$

For $f_1 = 1, f_2 = 0.5$, that's

$$V(\boldsymbol{\phi}_1) \approx V_0(0.882 + 0.470y).$$

For $f_1 = 1, f_2 = 2$, that's

$$V(\boldsymbol{\phi}_1) \approx V_0(0.135 + 0.991y).$$

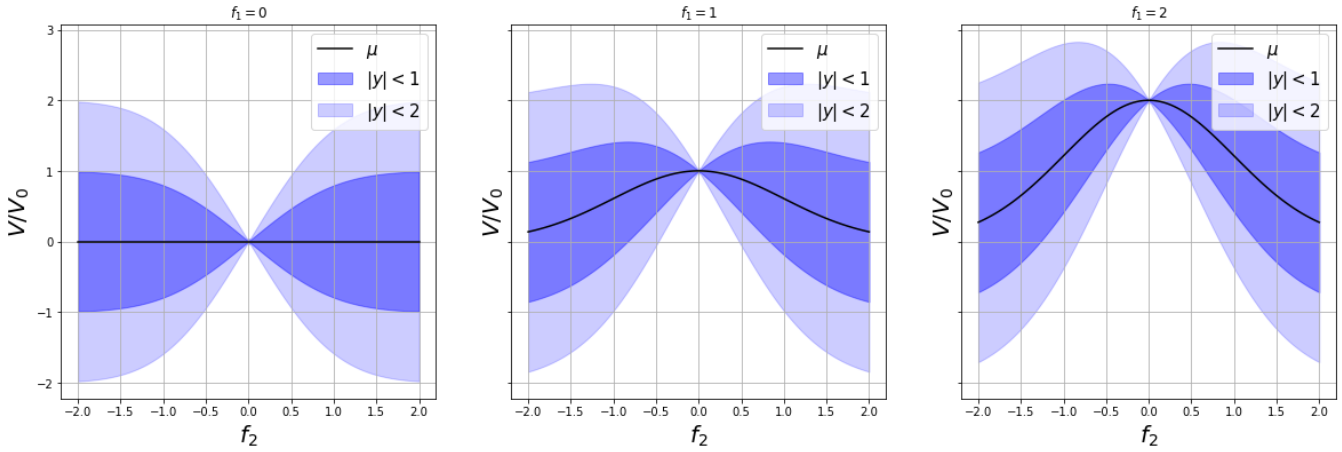

FIG. 2. One-two sigma contour illustrations of the correlation between two points as a function of their separation. Given $V(\phi_0^{(0)}) = f_1 V_0$ and simulating $V(\phi_0^{(0)} + f_2 s)$, the figure shows μ (black line) and the normal distributions for possible variation about μ. When $|f_2|$ is small, the evaluation points are close together and V will be highly constrained around $f_1 V_0$. When f_2 is large, V is very loosely constrained and will generate as if the original point is weakly influential.

Notice how μ (the first term) decreases from $f_1 V_0$ to 0 as f_2 increases, and how σ (the second term) increases from 0 to yV_0 at the same time. This behavior verifies the intuition that sampling $V(\boldsymbol{\phi}_1)$ far away from $\boldsymbol{\phi}_0$ should be similar to *unconstrained* sampling from a Normal distribution with mean 0, variance V_0^2, just like we did to find $V(\boldsymbol{\phi}_0)$. See Figure 2 to visualize this trend.

B. Generation: Many Points

When I ask to generate a third bit of data $V(\boldsymbol{\phi}_2)$, things would seem to get more complicated. After all, I will have to take into account both previous points. What if I have n data points and ask for point number $n+1$? What if I ask for multiple points at the same time? Thankfully, the methods described in the previous section generalize well to handle many points at once.

Imagine we have sampled n potential values at $\boldsymbol{\phi}_0, \boldsymbol{\phi}_1, \ldots, \boldsymbol{\phi}_{n-1}$ and we want to simulate p more values at $\boldsymbol{\phi}_n, \ldots, \boldsymbol{\phi}_{n+p-1}$. We can start again, with the covariance matrix $\boldsymbol{\Gamma}$, defined as in (27), which is $(n+p)\times(n+p)$ in this case. This time, we want a vector of p new correlated potential values as output, so let's call it $\boldsymbol{v}_N$, read "v-new", at the risk of confusion with N, the dimension of V. Correspondingly, call all the points we already know listed in a vector $\boldsymbol{v}_O$, for "v-old". All the points together in a vector, we can simply label $\boldsymbol{v}$. Some properties to note:

- $\boldsymbol{\Gamma} = \langle \boldsymbol{v}\boldsymbol{v}^T \rangle$, where $\boldsymbol{v}^T$ is a row vector.
- $\boldsymbol{\Gamma}$ is symmetric. Mathematically, $\Gamma^{(ij)} = \Gamma^{(ji)}$.
- $\boldsymbol{\Gamma}$ can be broken up into four blocks:

$$\boldsymbol{\Gamma} = \begin{bmatrix} \boldsymbol{\Gamma}_{OO} & \boldsymbol{\Gamma}_{ON} \\ \boldsymbol{\Gamma}_{NO} & \boldsymbol{\Gamma}_{NN} \end{bmatrix}$$

  - $\boldsymbol{\Gamma}_{OO} = \langle \boldsymbol{v}_O \boldsymbol{v}_O^T \rangle$ is the $n \times n$ upper-left block and contains all information correlating old points with old points.
  - $\boldsymbol{\Gamma}_{NO} = \boldsymbol{\Gamma}_{ON}^T = \langle \boldsymbol{v}_N \boldsymbol{v}_O^T \rangle$ are the $p \times n$ lower-left and $n \times p$ upper-right blocks, correlating old points with new points.
  - $\boldsymbol{\Gamma}_{NN} = \langle \boldsymbol{v}_N \boldsymbol{v}_N^T \rangle$ is the $p \times p$ lower-right block, correlating new points with new points.
- The current problem reduces to the problem in the previous section with $n = p = 1$.

Now, the new points will still be the sum of two predictions by $\boldsymbol{\Gamma}$. We can extend the scalar equation from the previous part to a vector equation, $\boldsymbol{v}_N = \boldsymbol{\mu} + \boldsymbol{\sigma}$, where $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are the expected potential value and the random correlated skew at *each* new point. Let's generalize the expressions for μ and σ to this multi-point problem. The predicted mean is

$$\boldsymbol{\mu} = \boldsymbol{\Gamma}_{NO} \boldsymbol{\Gamma}_{OO}^{-1} \boldsymbol{v}_O, \tag{28}$$

the conditional covariance is now a matrix,

$$\boldsymbol{\Gamma}_C = \boldsymbol{\Gamma}_{NN} - \boldsymbol{\Gamma}_{NO} \boldsymbol{\Gamma}_{OO}^{-1} \boldsymbol{\Gamma}_{ON}, \tag{29}$$

and the skew is

$$\boldsymbol{\sigma} = \boldsymbol{L}_C \boldsymbol{y} \tag{30}$$

where $\boldsymbol{\Gamma}_C = \boldsymbol{L}_C \boldsymbol{L}_C^T$ is the lower-triangular *Cholesky decomposition* of $\boldsymbol{\Gamma}_C$ and $\boldsymbol{y}$ is a vector of p randomly drawn real numbers from a standard normal distribution

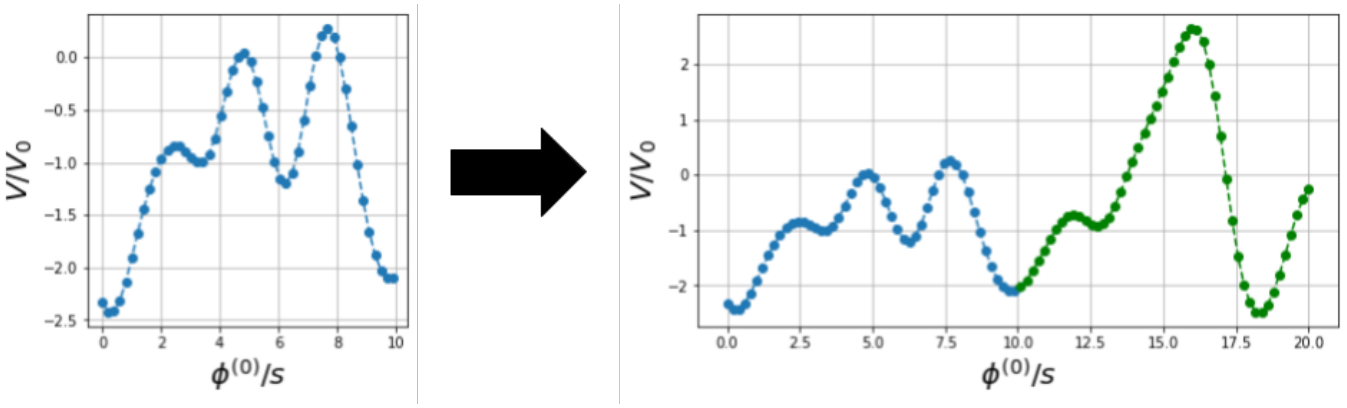

FIG. 3. A 1-dimensional Gaussian random potential generated many points at a time. Given fifty sampled potential values (blue), it is possible to generate fifty new potential values (green) all at once. The new values are appropriately constrained so as to form a continuous and differentiable curve with the old points.

[4]. These linear algebra techniques allow us to simulate any number of potential values constrained by any other number of values (see Figure 3). Mathematically, we say "$\boldsymbol{v}_N$ is drawn from a multivariate Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Gamma}_C$."

C. Generation: Derivatives of V

In addition to V, we need to randomly simulate first derivatives $\partial V/\partial\phi^{(\alpha)}$ and second derivatives $\partial^2 V/\partial\phi^{(\alpha)}\partial\phi^{(\beta)}$. No new linear algebra machinery is needed to do this. If I include values for derivatives of V inside the vector $\boldsymbol{v}$, then it is only necessary to populate $\boldsymbol{\Gamma}$ with the correct correlation functions. For instance,

$$\left\langle V(\boldsymbol{\phi}_i)\frac{\partial V}{\partial\phi_j^{(\alpha)}}(\boldsymbol{\phi}_j)\right\rangle = \frac{\partial}{\partial\phi_j^{(\alpha)}}\langle V(\boldsymbol{\phi}_i)V(\boldsymbol{\phi}_j)\rangle$$
$$= \left(\frac{\phi_i^{(\alpha)}-\phi_j^{(\alpha)}}{s^2}\right)\langle V(\boldsymbol{\phi}_i)V(\boldsymbol{\phi}_j)\rangle$$

where $\langle V(\boldsymbol{\phi}_i)V(\boldsymbol{\phi}_j)\rangle$ is given by (26). Correlations between derivatives and second derivatives can get tedious to calculate and implement as there are many cases for different indices of $\boldsymbol{\phi}$.

D. The Minimum Condition

In Section II B, I detailed necessary conditions on inflationary potentials so that they evolve $\boldsymbol{\phi}$ in physically meaningful ways. In the N-dimensional Gaussian random potential model, the slow-roll condition is satisfied for many choices of s, V_0, and N, but the minimum condition is *not* naturally satisfied. There is nothing inherent in the mathematics of the simulation that forces minima to spawn at exactly $V = 0$, as there was in the quadratic model. Observe that the $V(\boldsymbol{\phi})$ simulated in Figure 3 has extrema, but none are minima at exactly $V = 0$, nor will I ever find one. Thus, for this model to yield physically meaningful results, I must manually impose the minimum constraints before simulating the potential.

By definition of a minimum, I need to impose the following $N^2 + N + 1$ constraints on V.

$$V(\mathbf{0}) = 0 \tag{31}$$
$$\frac{\partial V}{\partial\phi^{(\alpha)}}(\mathbf{0}) = 0 \text{ for all } \alpha \tag{32}$$
$$\frac{\partial^2 V}{\partial(\phi^{(\alpha)})^2}(\mathbf{0}) > 0 \text{ for all } \alpha \tag{33}$$
$$\frac{\partial^2 V}{\partial\phi^{(\alpha)}\partial\phi^{(\beta)}}(\mathbf{0}) = 0 \text{ for all } \alpha \neq \beta \tag{34}$$

In words, we want the potential to be zero, flat, and concave up at the origin (I could have chosen any $\boldsymbol{\phi}$ as long as I was consistent). More precisely, we want the Hessian matrix $\boldsymbol{H}$ of second partials of V to be positive-definite—contain only positive eigenvalues—at the origin. Constraints (33) and (34) are equivalent to diagonalizing $\boldsymbol{H}$ at the origin and forcing the diagonal elements to be positive. Visually, we are exploiting the freedom to rotate our coordinate axes along the principal directions of curvature. This is a surefire method of enforcing minimum behavior in a local neighborhood, as long as the subsequently generated points are correlated properly. See Figure 4 to visualize the culmination of these efforts in a 2-dimensional simulation, visualized many different ways.

IV. COMPUTATIONAL PROCESS

In this section, I outline the computational methods and resources used to simulate V, evolve solutions, and test its predictions. The project is encompassed in an intricate yet efficient object-oriented Python program.

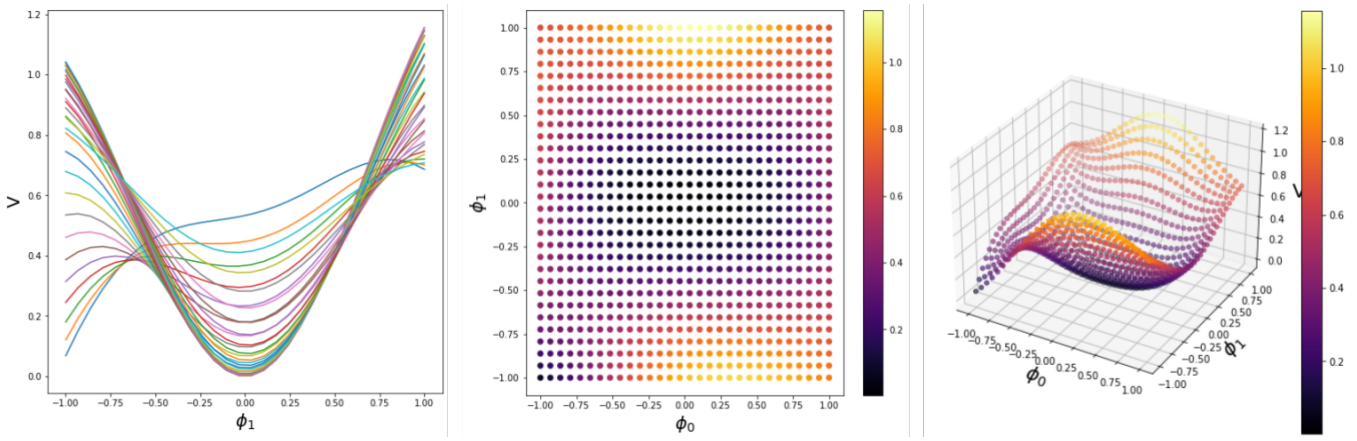

FIG. 4. A 2-dimensional Gaussian random potential generated at nodes on a 30×30 grid evenly spaced in $\phi^{(0)}, \phi^{(1)}$. The leftmost plot shows cross sections of V at constant values of $\phi^{(0)}$. The center plot shows a "bird's eye view" of the lattice with color indicating value for V. The rightmost plot is a traditional 3-d projection with axes $(\phi^{(0)}, \phi^{(1)}, V)$. Minimum conditions (31)—(34) were imposed before generating the 900 values. Solving the differential equation (8) for $\boldsymbol{\phi}$ under potentials like these will resemble a ball rolling around on this "landscape" under the influence of gravity and friction.

The code is all-original to this project, the product of over one thousand hours of effort. The relevant Python function within the code I have named `bushwhack`, since my solutions are found by numerical solving differential equations, step by step, in previously unknown, uncharted "landscapes". The `bushwhack` function takes many inputs and yields an object from an original class `BStuff`, which has dozens of attributes including information about the trajectory, observable quantities, and the computational process involved. All inputs and outputs relating to `bushwhack` are detailed in Appendix B. Recall that generating my Gaussian random potential is only the first step; I must solve for the evolution of $\boldsymbol{\phi}$ from its initial conditions until a stopping condition is met, and that differential equation is governed by my randomly generated V. After the trajectory is solved, I must extract from it observable quantities so that I can evaluate my model against Planck data and other theoretical constraints.

The code runs, roughly, in the following sequence.

- *Supply initial conditions.* The user supplies starting values of $\boldsymbol{\phi}$ and $d\boldsymbol{\phi}/dN_e$, as well as details about the potential, such as s, V_0, and N.
- *Force the minimum.* Seed the random generation of V with minimum conditions (31)—(34).
- *Generate V and $\nabla_{\boldsymbol{\phi}} V$ at $\boldsymbol{\phi}_i$.* At first, this is a point specified by the initial condition $\boldsymbol{\phi}_0$, but this step is repeated at other points.
- *Solve for $\boldsymbol{\phi}_{i+1}$.* Use equation (8) along with the V information generated at the previous point to numerically compute the next $\boldsymbol{\phi}$ value after one time step.
- *Repeat the previous two steps cyclically.* Cycle between generating the potential and solving the differential equation until the stopping condition $\epsilon \uparrow 1$ is met.
- *Check if the trajectory is plausible.* Perform elementary tests to confirm the $\boldsymbol{\phi}$ solution hit major checkpoints, like converging to the origin and accumulating enough e-folds of inflation.
- *Extract observable quantities.* If the trajectory is plausible, carry forward extracting predictions for perturbation spectra.

See Figure 5 for a flowchart. I will further detail each of these steps in the process below.

A. Initial Conditions

Initial conditions on the simulation are important knobs I can tweak to test which regions of "parameter space" statistically produce the most viable inflationary models. Now that we have seen Figure 4, we can visualize what each of s, V_0, and N physically change.

- Changing s changes the *width* of the landscape. Increasing s results in the hills and valleys spreading farther apart from one another.
- Changing V_0 changes the *height* of the landscape. Increasing V_0 makes the slopes steeper.
- Changing N changes the number of dimensions of V. V is a function from $\mathbb{R}^N \to \mathbb{R}$, so, as usual, our spatial axes available for visualization run out at $N = 2$.

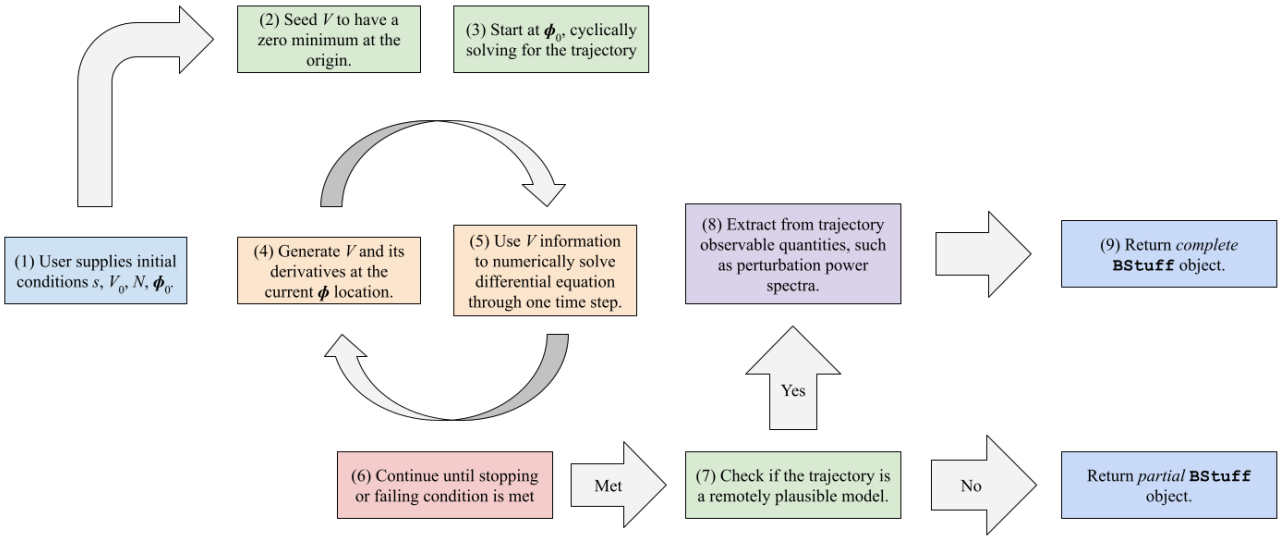

FIG. 5. Flowchart illustrating the chronology of the computational process involved in solving for the trajectory and extracting observable quantities. Each of these steps are thoroughly detailed in Section IV.

In addition to properties of the inflaton potential, the user has the option to place the inflaton field at any starting value $\boldsymbol{\phi}_0$ and at any starting velocity $d\boldsymbol{\phi}_0/dN_e$. There is no preferred axis among the N dimensions of V, so the only important quantity related to $\boldsymbol{\phi}_0$ is its distance from the imposed minimum. My minima will always be imposed at the origin so that this distance is just the magnitude $|\boldsymbol{\phi}_0| = \phi_0$. My initial value for $d\boldsymbol{\phi}/dN_e$ is interesting because I assign it after the potential information at $\boldsymbol{\phi}_0$ has been generated. Specifically, I want the inflaton field to start out at slow roll, an equilibrium or terminal velocity that balances the frictional term in (8) with the slope term. This is found most easily by using (6) and (10). Since we want the acceleration of the inflaton field to be zero during slow roll, we set $\ddot{\boldsymbol{\phi}} \approx 0$. Now,

$$\dot{\boldsymbol{\phi}} \approx -\frac{\nabla_{\boldsymbol{\phi}} V}{3H} \tag{35}$$

Assuming $|\dot{\boldsymbol{\phi}}|^2$ is negligibly small as well, (10) reduces to $3H^2 \approx V$. Substituting and rearranging, we have

$$\frac{1}{H}\frac{d\boldsymbol{\phi}}{dt} \approx -\frac{\nabla_{\boldsymbol{\phi}} V}{V}. \tag{36}$$

Finally, we observe that

$$\frac{dN_e}{dt} = \frac{d(\ln a)}{dt} = \frac{1}{a}\frac{da}{dt} = H, \tag{37}$$

which resolves the left-hand side of (36) to $d\boldsymbol{\phi}/dN_e$. So, once the first values of the potential in the trajectory are known, the initial velocity condition can be set to slow roll by (36).

B. Force the Minimum

Before simulating any information about V, we need to impose conditions (31)—(34) to ensure the subsequent generation can plausibly represent our universe. I manually impose conditions (31), (32), and (34), initializing arrays such as $\boldsymbol{\Gamma}$, $\boldsymbol{v}$, $\boldsymbol{\phi}$, and $d\boldsymbol{\phi}/dN_e$. Condition (33) is imposed by randomly simulating N independent second derivative values, taking their absolute values to force them positive, and storing them as the diagonals of the Hessian of V at the origin. These values will remain the first $N^2 + N + 1$ entries of $\boldsymbol{v}$ throughout the entire simulation, propagating into the upper-left $(N^2 + N + 1) \times (N^2 + N + 1)$ block of $\boldsymbol{\Gamma} = \langle \boldsymbol{v}\boldsymbol{v}^T \rangle$ as constraints on the rest of the potential.

C. Simulate the Potential

Now that I have forced the minimum, I can begin to simulate correlated potential values near the minimum. Importantly, I will only generate potential values exactly *on the path evolution of* $\boldsymbol{\phi}$, not in the surrounding regions. The potential in the regions near the path does not influence equations of motion nor is it relevant when extracting observable quantities. This is a time saving technique, as it is easier to perform linear algebra with smaller $\boldsymbol{\Gamma}$ (more in Section V).

I start with the user-supplied initial condition $\boldsymbol{\phi}(0) = \boldsymbol{\phi}_0$, generating $V(\boldsymbol{\phi}_0)$ and $\nabla_{\boldsymbol{\phi}} V(\boldsymbol{\phi}_0)$. This is done, for

the initial point in the trajectory as well as for every other point, exactly as detailed in Section III B with $n = \texttt{length}(\boldsymbol{v}_O)$ and $p = N + 1$. The covariance matrix $\boldsymbol{\Gamma}$ is broken up into blocks and the newly generated potential values are constrained, via linear algebra, by the old potential values which are securely stored in $\boldsymbol{v}_O$. Once the trajectory is complete, both V and $\nabla_{\boldsymbol{\phi}} V$ will be known at every value of $\boldsymbol{\phi}$ along the path.

D. Solve the Differential Equation

Let's return to the beginning of the trajectory for a moment. With $V(\boldsymbol{\phi}_0)$ and $\nabla_{\boldsymbol{\phi}} V(\boldsymbol{\phi}_0)$ simulated properly, we can numerically solve (8) for incremental inflaton values $\boldsymbol{\phi}_1$ and $d\boldsymbol{\phi}_1/dN_e$ after some time step ΔN_e. Euler's method, the most elementary numerical differential equation solving technique, would equate

$$\begin{bmatrix} \boldsymbol{\phi}_1 \\ d\boldsymbol{\phi}_1/dN_e \end{bmatrix} \approx \begin{bmatrix} \boldsymbol{\phi}_0 \\ d\boldsymbol{\phi}_0/dN_e \end{bmatrix} + \begin{bmatrix} d\boldsymbol{\phi}_0/dN_e \\ d^2\boldsymbol{\phi}_0/dN_e^2 \end{bmatrix} \Delta N_e, \quad (38)$$

where $d^2\boldsymbol{\phi}_0/dN_e^2$ is found by substituting known values into (8) and solving for the first term. I used the more accurate numerical technique "fourth-order Runge-Kutta", which takes into account information from many previous time steps to predict the next inflaton field value. The solution technique is manipulable via the `method` parameter in the `bushwhack` function, but Runge-Kutta is adequate for most relevant circumstances. Just like I used to numerically find solutions with the quadratic potential, the code for these numerical techniques is attributable to the `solve_ivp` function of the `scipy` module.

E. Repeat

Steps C and D are repeated over and over again, one time step at a time, storing inflaton field and potential information at each point. Additionally, $\epsilon = \frac{1}{2}|d\boldsymbol{\phi}/dN_e|^2$ is monitored at every step. It turns out that for plausible inflationary models, ϵ starts between 0 and 1 and rapidly increases through 1 as $\boldsymbol{\phi}$ makes its final acceleration off the slopes of V into the minimum at 0. This makes $\epsilon \uparrow 1$ a perfect stopping criterion, since all physically meaningful behavior occurs when V is significantly greater than 0 (see Figure 1 to recall damped oscillatory behavior when $\boldsymbol{\phi}$ settled in the quadratic minimum). Figure 6 gives an example of a usual progression of ϵ.

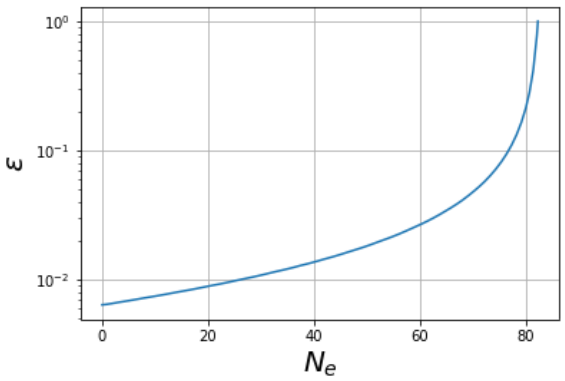

FIG. 6. One representative $\epsilon(N_e)$, a stopping criterion for numerically solving the differential equation (8). Notice how ϵ rapidly increases to 1 around $N_e = 80$, signaling the end of inflation.

There are some stopping criteria in addition to ϵ that help to end simulations that are going awry. The following quantities are monitored at every time step, just like ϵ, in an effort to identify solutions that are not worth integrating further.

- *Inwardness.* I have assigned the lowercase Greek letter iota to measure how much the slopes of V "point" toward the minimum at the origin, the "inwardness":

$$\iota(\boldsymbol{\phi}) = \nabla_{\hat{\boldsymbol{\phi}}} V \cdot \hat{\boldsymbol{\phi}} \quad (39)$$

  Overhats indicate unit vectors. When $\iota \approx 1$, the slopes of V point directly downhill toward the origin, and thus the inflaton field is expected to accelerate in that direction. If $\iota < 0$, the inflaton field is likely to diverge away from the minimum at the origin, leading to an unrealistic evolution. So, if $\boldsymbol{\phi}$ evolves into a region where $\iota < 0$, I admit that the simulation failed.

- *Altitude.* Another simple indication of failure is if $V(\boldsymbol{\phi}) < 0$ at any point. It is unlikely that this fail criterion would be triggered before the inwardness criterion, but if V falls below zero, the cycle will immediately halt.

Failing criteria are of critical importance when testing initial values $\boldsymbol{\phi}_0$ far away from the origin. The farther away $\boldsymbol{\phi}$ starts from the origin, the more likely it is that the solution will diverge to another minimum in the random potential. The program relies on failing criteria to identify divergent behavior before the solver wastes too much time or crashes by taking square roots of negative quantities.

F. Plausibility Check

Once the iteration has ended, either by $\epsilon \uparrow 1$ or a failing criterion, the code performs some quick checks to judge whether to perform further analysis on the trajectory. The plausibility checks are as follows.

1. Did the solution fail by diverging? In other words, was $\iota(\boldsymbol{\phi}_f) < 0$? If so, do not analyze further. If not, proceed to the next check.

2. Did the solution fail by submerging? In other words, was $V(\boldsymbol{\phi}_f) < 0$? If so, do not analyze further. If not, proceed to the next check.

3. Did the solution accumulate enough inflation? In other words, was $N_e(t_f) \geq 55$? If so, this solution is plausible enough and observable quantities can be extracted. If not, do not analyze further.

This step is very quick. Even if a solution does not pass the plausibility checks, the `bushwhack` function still outputs a `BStuff` object with trajectory attributes so that the failure may be examined or recorded.

G. Extract Observables

Finally, properties of tensor and scalar perturbation spectra can be extracted from the trajectory via methods described in Section II D and Appendix A. With each trajectory that passes the plausibility checks, there is an associated $A_S, n_S, A_T, n_T, r, A_{iso}$, and n_{iso}. Even with the same set of parameters and initial conditions, randomized potentials will yield different trajectories, which will yield different values for these observable quantities. It is only with many repeated trials that the program can state the statistical predictions of a particular set of parameters.

There are many different variables and symbols associated with the trajectory of $\boldsymbol{\phi}$ and its extracted observable quantities; it can be overwhelming. See Figure 7 to visualize all the different ideas side-by-side for one specific 2-dimensional trajectory.

V. DATA COMPRESSION AND EFFICIENCY

The covariance matrix $\boldsymbol{\Gamma}$ of constraints on the potential can grow very large, very quickly. With $N+1$ rows and columns added to the matrix at every time step, $\boldsymbol{\Gamma}$ can easily grow to contain millions of elements before the simulation finishes. This reduces efficiency when populating new rows and columns of the matrix and increases the chance for numerical instabilities during linear algebra steps such as inverses and Cholesky decompositions. Several measures were implemented to ease this computational intensity during the simulate-solve cycle.

A. Predicting Plausibility

No matter how well-optimized the simulate-solve cycle becomes, we never want to waste time finding solutions that are destined to be implausible from the beginning. If there is a way to perform the sort of plausibility check detailed in Section IV F before the simulation even starts, it could save lots of time, especially when I need to perform as many simulations as I can.

It turns out we can estimate the total number of e-folds of inflation a trajectory will yield with just the very first simulated potential values $V(\boldsymbol{\phi}_0)$ and $\nabla_{\boldsymbol{\phi}} V(\boldsymbol{\phi}_0)$. In the slow roll approximation, equation (36) can be solved for dN_e.

$$dN_e \approx -\frac{V}{\nabla_{\boldsymbol{\phi}} V} d\boldsymbol{\phi} \tag{40}$$

If $V = V(\phi)$ is a function of one variable, the number of e-folds accumulated over a portion of the trajectory can be approximated by an integral.

$$N_e \approx -\int_{\phi_i}^{\phi_f} \frac{V(\phi)}{V'(\phi)} d\phi = \int_{\phi_f}^{\phi_i} \frac{V}{V'} d\phi \tag{41}$$

Near the minimum, the Gaussian random potential can be approximated to a quadratic $V = k\phi^2$. Also, we intend to integrate over the entire trajectory, from the initial point $\phi_i = \phi_0$ to the final point $\phi_f = 0$. Substituting, that is

$$N_e \approx \frac{1}{2} \int_0^{\phi_0} \phi \; d\phi = \frac{\phi_0^2}{4}. \tag{42}$$

This is a fairly good approximation, but we can actually improve by allowing $V = k\phi^m$ for some constant m, instead of forcing $m = 2$ in the quadratic approximation. In this approximation, $V/V' = \phi/m$, thus initial conditions yield $m = (V'(\phi_0)/V(\phi_0))\phi_0$, and the integral becomes

$$N_e \approx \frac{1}{\phi_0} \frac{V(\phi_0)}{V'(\phi_0)} \int_0^{\phi_0} \phi \; d\phi = \frac{\phi_0}{2} \frac{V(\phi_0)}{V'(\phi_0)}. \tag{43}$$

This approximation generalizes naturally back to higher dimensions of V.

$$N_e \approx \frac{|\boldsymbol{\phi}_0|}{2} \frac{V(\boldsymbol{\phi}_0)}{|\nabla_{\boldsymbol{\phi}} V(\boldsymbol{\phi}_0)|} \tag{44}$$

For instance, in the simulation visible in Figure 7, the number of e-folds of inflation totaled $N_e = 182.32$. At the beginning of the simulation, the quadratic approximation (eq. (42)) for V would have predicted a total of $N_e = 204.62$ (standard error of about 10.9%), while the more general approximation (43) predicted $N_e = 188.15$ (standard error of 3.1%).

After the very first potential values in the trajectory are simulated, I use equation (44) to estimate the total number of e-folds the run will accumulate. If the estimation is below a lower bound, I choose to discard the entire run. I set the bound generously low at $(N_e)_{LB} = 45$ because the estimation is not always very accurate. However, this cutoff saves a lot of time when experimenting with initial conditions $\boldsymbol{\phi}_0$ close to the origin, since it efficiently isolates the few runs that will accumulate enough inflation to be reasonable.

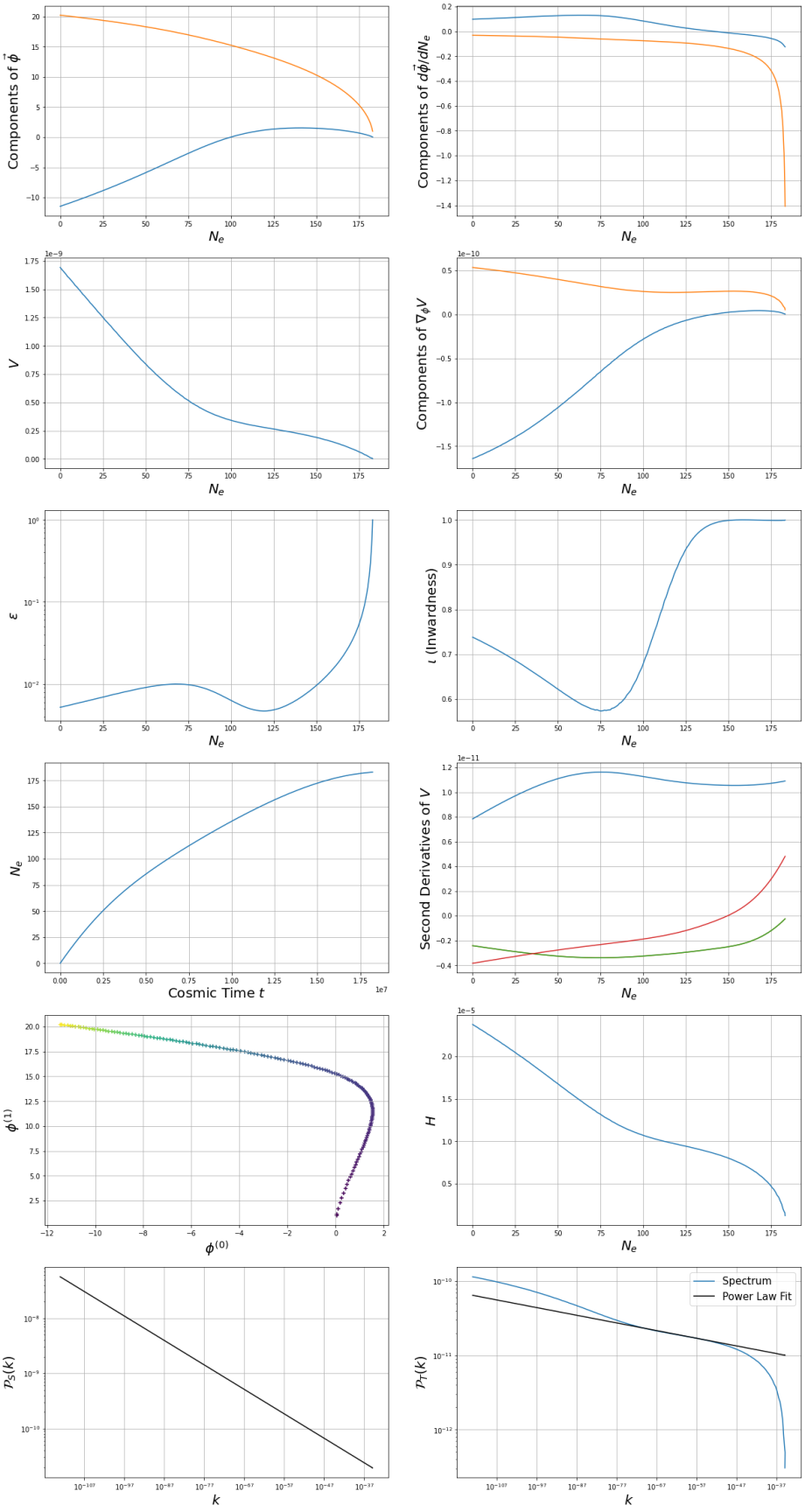

FIG. 7. Visualizations of different quantities related to the trajectory. Every one of these plots contains information about the *same* trajectory, simulated under a Gaussian random potential with $s = 30$, $V_0 = 5 \cdot 10^{-9}$, and $N = 2$. Row 1 contains information about $\boldsymbol{\phi}$ over time, row 2: $V(\boldsymbol{\phi})$ over time, row 3: stopping and failing conditions over time, row 4: inflation and second derivatives of V over time, row 5: $(\phi^{(0)}, \phi^{(1)})$ plot and Hubble parameter, row 6: perturbation spectra.

B. Simulating Along the Path

The number of potential data points that need to be stored in $\boldsymbol{v}$, and thus the number of constraints within $\boldsymbol{\Gamma}$, are greatly reduced by simulating potentially along the only relevant curve in $\boldsymbol{\phi}$-space: the solution curve. As mentioned previously, the only potential information that is relevant to the progression of the trajectory and the extracted observable quantities are the values encountered along the path evolution of $\boldsymbol{\phi}$. It would be a waste of time to simulate elsewhere.

Second derivatives of the potential $\partial^2 V/\partial\phi^{(\alpha)}\partial\phi^{(\beta)}$ are not simulated during the trajectory because they are only necessary in the calculation of the scalar perturbation power spectrum (Appendix A.). However, if a certain run passes all the plausibility tests, I need to return to $\boldsymbol{\Gamma}$ to simulate second derivatives along the path. This is a computationally intensive step for a few reasons:

- $\boldsymbol{\Gamma}$ is already large because it encompasses all of the information about the trajectory.
- There are $(N^2+N)/2$ independent second derivatives, including mixed partials, to be calculated all along the path. When N gets large, this quickly becomes intractable.
- The sorting and length of formulas involved in applying the correct correlation functions to second derivatives is considerably more computationally difficult than correlating zeroeth and first derivatives. Recall, every new entry in $\boldsymbol{\Gamma}$ is computed via a correlation function.

I reduce computational intensity at this step by simulating only the independent second derivatives (either the lower or upper triangle of the Hessian $\boldsymbol{H}$) at sparsely spaced points throughout the trajectory. Often times, the trajectory yields discrete points that are dense in $\boldsymbol{\phi}$-space, unnecessarily dense for the purposes of second derivatives. I can sample much more sparsely (say, 10 times instead of 500) and perform `CubicSpline` interpolation to fill the gaps reasonably.

C. Avoiding Matrix Inverses

Equations (28) and (29) reveal that it is necessary to take inverses of $\boldsymbol{\Gamma}_{OO}$ in order to properly simulate new potential values constrained by old ones. This is not a problem in theory; $\boldsymbol{\Gamma}$, and by extension $\boldsymbol{\Gamma}_{OO}$, is positive-definite and thus invertible no matter what. Computationally, taking inverses of these matrices can result in nightmarish errors. In almost every simulation, potential data is simulated and stored that yield relatively redundant constraints. Redundancy is quantifiable by examining the eigenvalues of $\boldsymbol{\Gamma}$, which rank the importance, in a sense, of each constraint. Two points of potential data that contain almost exactly the same information (because, for instance, they are evaluated at very proximate values) would appear in the spectrum of $\boldsymbol{\Gamma}$ as one moderate eigenvalue and one very small eigenvalue. Matrices with *very* small eigenvalues are called *ill-conditioned*, well known to data scientists. Ill-conditioned matrices cause numerical instabilities in both matrix inversion and Cholesky decomposition, two vital steps in generating potential values, so we need to deal with them effectively. We employ two techniques detailed below.

- *Add noise to* $\boldsymbol{\Gamma}$. As careless as it may sound, it is possible to surpass many numerical instabilities in both inversion and Cholesky decomposition by adding small multiples of the identity to $\boldsymbol{\Gamma}$ momentarily while the computation is done. In other words, let

$$\tilde{\boldsymbol{\Gamma}} = \boldsymbol{\Gamma} + \tilde{n}\mathbb{I}_{\mathtt{length}(\boldsymbol{\Gamma})} \tag{45}$$

  where $\tilde{n}$ is called *noise* and $\mathbb{I}_n$ is the $n \times n$ identity. If $\tilde{n}$ is sufficiently small, it can resolve numerical instabilities while contributing a vanishingly small influence on the simulation of new points. In practice, $\tilde{n}$ must be increased to a surprisingly high fraction of the highest eigenvalues of $\boldsymbol{\Gamma}$ before any peculiarities in the simulation are noticeable. In my code, I determined that an optimal value for $\tilde{n}$ is a minute 10^{-12}. Even that value is conservatively high so as to avoid crashes at any point. The un-noised version of $\boldsymbol{\Gamma}$ is the only matrix that is stored; noise is only added when it comes time to perform linear algebra. This solution completely satisfies Cholesky decompositions, but inversion is still prone to yield heart-wrenching numerical errors without implementing the next fix.

- *Rephrase matrix equations.* Often times, matrix equations can be rewritten in equivalent forms that seem identical to humans, but make a world of difference to computers. For instance, if I want $\boldsymbol{x} = \boldsymbol{A}^{-1}\boldsymbol{y}$, I can find this by inverting $\boldsymbol{A}$ and performing matrix multiplication, or I can rephrase the equation as $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}$ and solve the system of equations for $\boldsymbol{x}$. For computers, the latter option is often much more reliable.

  Equations (28) and (29) are the problematic ones, so we seek to rephrase them in efficient ways. First, Cholesky decompose $\boldsymbol{\Gamma}_{OO}$ into

$$\boldsymbol{\Gamma}_{OO} = \boldsymbol{L}_{OO}\boldsymbol{L}_{OO}^T. \tag{46}$$

  The matrix $\boldsymbol{L}_{OO}$ is lower-triangular, making it exceptionally versatile and efficient when solving linear systems. Exploiting that property, I solve the matrix equation

$$\boldsymbol{L}_{OO}(\boldsymbol{L}_{OO}^{-1}\boldsymbol{\Gamma}_{ON}) = \boldsymbol{\Gamma}_{ON} \tag{47}$$

for $\boldsymbol{L}_{OO}^{-1}\boldsymbol{\Gamma}_{ON}$ using the phenomenal `solve_triangular` function within the `scipy.linalg` module. Now, (29) becomes

$$\begin{aligned}\boldsymbol{\sigma} &= \boldsymbol{\Gamma}_{NN} - (\boldsymbol{L}_{OO}^{-1}\boldsymbol{\Gamma}_{ON})^T(\boldsymbol{L}_{OO}^{-1}\boldsymbol{\Gamma}_{ON}) \\ &= \boldsymbol{\Gamma}_{NN} - \boldsymbol{\Gamma}_{NO}(\boldsymbol{L}_{OO}^{-1})^T\boldsymbol{L}_{OO}^{-1}\boldsymbol{\Gamma}_{ON} \\ &= \boldsymbol{\Gamma}_{NN} - \boldsymbol{\Gamma}_{NO}(\boldsymbol{L}_{OO}\boldsymbol{L}_{OO}^T)^{-1}\boldsymbol{\Gamma}_{ON} \\ &= \boldsymbol{\Gamma}_{NN} - \boldsymbol{\Gamma}_{NO}\boldsymbol{\Gamma}_{OO}^{-1}\boldsymbol{\Gamma}_{ON}.\end{aligned} \tag{48}$$

Also, I solve a separate matrix equation,

$$\boldsymbol{L}_{OO}(\boldsymbol{L}_{OO}^{-1}\boldsymbol{v}_O) = \boldsymbol{v}_O, \tag{49}$$

for $\boldsymbol{L}_{OO}^{-1}\boldsymbol{v}_O$ so that the predicted mean in (28) is

$$\begin{aligned}\boldsymbol{\mu} &= (\boldsymbol{L}_{OO}^{-1}\boldsymbol{\Gamma}_{ON})^T(\boldsymbol{L}_{OO}^{-1}\boldsymbol{v}_O) \\ &= \boldsymbol{\Gamma}_{NO}(\boldsymbol{L}_{OO}\boldsymbol{L}_{OO}^T)^{-1}\boldsymbol{v}_O \\ &= \boldsymbol{\Gamma}_{NO}\boldsymbol{\Gamma}_{OO}^{-1}\boldsymbol{v}_O\end{aligned} \tag{50}$$

Now, I have developed clever tricks to bypass all the inverses required to simulate new potential points. In the mean time, I have ensured that the process remains numerically stable.

D. Matrix Updates

As $\boldsymbol{\Gamma}$ and its associated matrices grow large, I need to avoid recalculating $\langle \boldsymbol{v}\boldsymbol{v}^T \rangle$ as much as I can. This means that I populate each incrementally larger covariance matrix in a vectorized manner. If $\texttt{length}(\boldsymbol{v}_O) = n$ and $\texttt{length}(\boldsymbol{v}_n) = p$, then updating $\boldsymbol{\Gamma}_i$ to $\boldsymbol{\Gamma}_{i+1}$ proceeds as follows.

1. Create an $(n+p) \times (n+p)$ matrix of zeros. This looks like

   `newGamma = zeros((n+p, n+p))`.

2. Populate the upper-left $n \times n$ block with $\boldsymbol{\Gamma}_i$ all at once. This looks like

   `newGamma[:n, :n] = Gamma`.

3. Populate the rest of the new matrix with appropriate correlation functions. Vectorizing this step involves populating every element that requires the same correlation function at the same time, involving extensive, tedious coding.

Another matrix that needs to be updated incrementally is the Cholesky decomposition $\boldsymbol{L}$ of $\boldsymbol{\Gamma}$. This is a common problem for data scientists and the solution is a *Rank 1 Cholesky update*. If $\boldsymbol{L}_i = \boldsymbol{L}_{OO}$ is the Cholesky decomposition of $\boldsymbol{\Gamma}_{OO}$ at some time step, then $\boldsymbol{L}_{i+1} = \boldsymbol{L} = \text{chol}(\boldsymbol{\Gamma})$ can be expressed as

$$\boldsymbol{L} = \begin{bmatrix} \boldsymbol{L}_{OO} & \boldsymbol{0} \\ \boldsymbol{L}_{OO}^{-1}\boldsymbol{\Gamma}_{ON} & \text{chol}(\boldsymbol{\Gamma}_C). \end{bmatrix} \tag{51}$$

This update allows us to use the previous Cholesky decomposition to populate the majority of a current decomposition, then perform less computationally intensive linear algebra to populate the remainder of the lower triangle.

E. Unprincipled Forgetting

Even with all the previous optimizations, $\boldsymbol{\Gamma}$ can become large and cumbersome, particularly apparent when populating new rows and columns. To refresh, we have previous information about the potential stored in $\boldsymbol{v}_O$ and we want future information about the potential, $\boldsymbol{v}_N$. If the length of $\boldsymbol{v}_O$ is $(N+1)n$, for $N+1$ data points at n time steps, and I need to populate $N+1$ more rows and columns for $\boldsymbol{v}_N$, that is

$$\begin{aligned}\text{new entries} &= (N+1)^2(n+1)^2 - (N+1)^2n^2 \\ &= (N+1)^2(2n+1)\end{aligned}$$

more entries of $\boldsymbol{\Gamma}$ that need to be filled. With $N = 3$ and $n = 200$, a conservative example likely near the middle of the simulation, that is 6416 entries, or over 3200 correlation functions that need to be computed. That number only increases, too.

Thankfully, there are ways to further ease the computational heftiness of $\boldsymbol{\Gamma}$. I can execute "forgetting steps" when the matrix reaches a predetermined upper limit on size to reduce its number of rows and columns while retaining its most important constraints. There are two methods of forgetting that I have implemented within the simulate-solve cycle, methods that we have lovingly named "principled" and "unprincipled" forgetting.

Unprincipled forgetting is temporarily removing rows and columns of $\boldsymbol{\Gamma}$ that correspond to older simulated points stored in $\boldsymbol{v}_O$. Points that were simulated toward the beginning of the trajectory may be relatively far away in $\boldsymbol{\phi}$-space from new points that need to be simulated in a different region. If I need to simulate $V(\boldsymbol{\phi}_{200})$, equation (26) tells us that the importance of previously generated potential data points depends on how far away they were generated from this new point, $\boldsymbol{\phi}_{200}$. In a continuous trajectory, the most recent points $(V(\boldsymbol{\phi}_{199}), V(\boldsymbol{\phi}_{198})$, etc.) are the most important in this generation since they are the most proximate. Potential data generated in the beginning of the trajectory $(V(\boldsymbol{\phi}_0), V(\boldsymbol{\phi}_1)$, etc.) will have much less influence. As it turns out, the constraining power of $\boldsymbol{\Gamma}$ is almost entirely contained in its data generated closest to the new point of evaluation. The "constraining power" of $\boldsymbol{\Gamma}$ is revealed by $\boldsymbol{\Gamma}_C$, which determines the scale of $\boldsymbol{\sigma}$ (eq. (30)), and thus how much a newly sampled potential point can vary. We can justify that removing rows and columns of $\boldsymbol{\Gamma}$ can have negligible impact on constraints with an example.

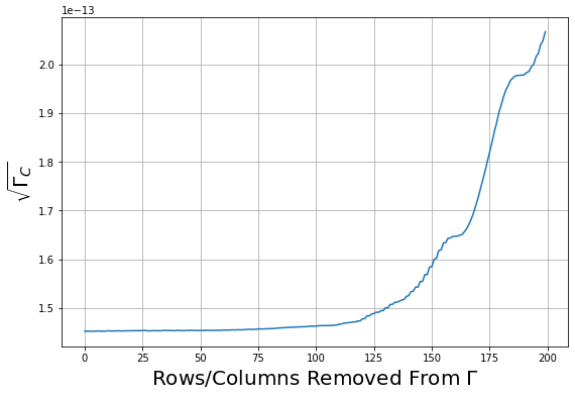

FIG. 8. Constraints on the value of a simulated point depend on how much previous information is incorporated into the calculation. As more information is removed (higher values on the horizontal axis), the constraints become more relaxed (Γ_C increases). However, the constraining power of $\mathbf{\Gamma}$ is retained even with many rows and columns removed.

Take the simulation shown in (7) and imagine a covariance matrix of constraints from only V along the trajectory, not its including derivatives or data at the minimum. That is, $\boldsymbol{v}_O$ is made up entirely of numbers that look like $V(\boldsymbol{\phi}_i)$, and $\mathbf{\Gamma}_{OO} = \langle \boldsymbol{v}_O \boldsymbol{v}_O^T \rangle$. Imagine I want to simulate a new potential value at the origin constrained by this partial covariance matrix. $\mathbf{\Gamma}_{OO}$ contains all information about the trajectory, but the data in the later rows and columns are most important to this simulation since it was generated closer to $\mathbf{0}$. $\mathbf{\Gamma}_C = \Gamma_C$ is a scalar in this case since we are only generating one constrained random number. We can calculate Γ_C with $\mathbf{\Gamma}_{OO}$, and that will give us an idea of how constrained the new potential value will be. We can then begin removing the oldest rows and columns of $\mathbf{\Gamma}$ and recalculating Γ_C to see if that constraint changes. The result is visualized in Figure 8.

As more information is removed from the simulation of $V(\mathbf{0})$, the constraints Γ_C become more relaxed, as expected. However, the difference between the constraining power of $\mathbf{\Gamma}$ when all information is incorporated (rows/columns removed = 0 in Figure 8) to when 100 rows/columns are removed is negligible. In this example, $\mathbf{\Gamma}$ is only approximately 300×300, so removing 100 rows and columns reduces the size of $\mathbf{\Gamma}$ by about 55 percent.

Unprincipled forgetting, done properly, is inconsequential to the constraining power of the covariance matrix, but highly effective in reducing computational intensity. In fact, unprincipled forgetting—ignoring rows and columns of $\mathbf{\Gamma}$ during simulation—can be performed multiple times over the course of a run to keep $\mathbf{\Gamma}$ at a manageable size. Even though newly simulated potential along the path will only be constrained by its closer neighbors (and the minimum), in practice there is negligible difference.

F. Principled Forgetting

Principled forgetting uses linear algebra and data compression techniques to algorithmically compress the covariance matrix $\mathbf{\Gamma}$ in a way that maximizes its constraining power while minimizing its overall size. When $\boldsymbol{v}_O$ gets large and redundant, we seek to find a compression matrix $\boldsymbol{A}$ such that

$$\tilde{\boldsymbol{v}}_O = \boldsymbol{A}\boldsymbol{v}_O, \tag{52}$$

where $\tilde{\boldsymbol{v}}_O$ is a smaller, compressed version of $\boldsymbol{v}_O$. In this case, we can define other compressed matrices.

$$\tilde{\mathbf{\Gamma}}_{OO} = \langle \tilde{\boldsymbol{v}}_O \tilde{\boldsymbol{v}}_O^T \rangle = \boldsymbol{A}\mathbf{\Gamma}_{OO}\boldsymbol{A}^T \tag{53}$$

$$\tilde{\mathbf{\Gamma}}_{NO} = \langle \tilde{\boldsymbol{v}}_N \tilde{\boldsymbol{v}}_O^T \rangle = \boldsymbol{A}\mathbf{\Gamma}_{NO}\boldsymbol{A}^T \tag{54}$$

$$\tilde{\mathbf{\Gamma}}_C = \mathbf{\Gamma}_{NN} - \tilde{\mathbf{\Gamma}}_{NO}\tilde{\mathbf{\Gamma}}_{OO}^{-1}\tilde{\mathbf{\Gamma}}_{ON} \tag{55}$$

The goal in specifying $\boldsymbol{A}$ is to keep $\tilde{\mathbf{\Gamma}}_C$ as close to $\mathbf{\Gamma}_C$ as possible, information-wise. Information about the constraints from $\boldsymbol{v}_O$ on $\boldsymbol{v}_N$ are found in the eigenvalues of the second term of $\tilde{\mathbf{\Gamma}}_C$. Mathematically, we seek to maximize the trace of that term:

$$\tau = \mathrm{Tr}(\tilde{\mathbf{\Gamma}}_{NO}\tilde{\mathbf{\Gamma}}_{OO}^{-1}\tilde{\mathbf{\Gamma}}_{ON}) \tag{56}$$

$$= \mathrm{Tr}\left(\mathbf{\Gamma}_{NO}\boldsymbol{A}^T(\boldsymbol{A}\mathbf{\Gamma}_{OO}\boldsymbol{A}^T)^{-1}\mathbf{\Gamma}_{ON}\right). \tag{57}$$

This constrained maximization problem, with some clever linear algebra tricks, is reformulated as a generalized eigenvalue problem.

$$\mathbf{\Gamma}_{ON}\mathbf{\Gamma}_{NO}\boldsymbol{a} = \lambda\mathbf{\Gamma}_{OO}\boldsymbol{a}, \tag{58}$$

The eigenvalues λ and eigenvectors $\boldsymbol{a}$ calculable via (58) are exactly the eigenvalues and vectors of the matrix in (56). Since we seek to maximize τ, we rank the eigenvalues λ from highest to lowest and populate the rows of $\boldsymbol{A}$ with the eigenvectors $\boldsymbol{a}$ corresponding to the highest ones. We can choose to populate $\boldsymbol{A}$ with as many rows as we want, but it is a trade-off between accuracy and efficiency. If we populate $\boldsymbol{A}$ with many eigenvectors, the trace τ will be very close to the uncompressed trace, but the covariance matrix won't be compressed very much. If we only populate $\boldsymbol{A}$ with a few eigenvectors, we can achieve staggering compression, but we risk forgetting too much information.

For the majority of trajectories, it is possible to keep an astounding portion of the trace—up to one part in a hundred million—while compressing the covariance matrix over 99.9%. Algorithmic compression, or principled forgetting, yields far better compression than simply ignoring earlier rows and columns, the method I called unprincipled forgetting.

I encountered two problems that left me unable to implement this algorithmic compression method efficiently:

1. $\mathbf{\Gamma}_{ON}\mathbf{\Gamma}_{NO}$ turns out to be more complicated than just multiplying two matrices; its elements must be found by *very* lengthy, tedious formulas, found by integrating the product of *two* correlation functions. I already needed to take care with indices when correlating derivatives of V, and now elements of this matrix demand that I manage formulas with *twice* as many indices. The number of independent formulas necessary to properly populate this matrix is overwhelming, but not totally impossible to handle, in theory.

2. Even if I write down and implement all formulas for $\mathbf{\Gamma}_{ON}\mathbf{\Gamma}_{NO}$, as I did up through the first derivatives of V, I actually have to compute the matrix when the compression steps come. This involves populating a matrix that is just as big as $\mathbf{\Gamma}_{OO}$, losing any time we hoped to gain after compressing $\mathbf{\Gamma}$. Updating the matrix incrementally, as I have done with $\mathbf{\Gamma}$ and $\boldsymbol{L}$, does little to help.

Algorithmic compression was implemented at one point, but is currently removed from my simulate-solve cycle for the reasons stated above, so it is possible to improve the rigor with which I optimize the program. However, unprincipled forgetting has propelled the code to a level of efficiency adequate for performing high numbers of simulations in $N \leq 3$ dimensions in reasonable time (more in Section VI).

G. Code Parallelization

For a given set of parameters and initial conditions, I need to simulate large numbers of different random paths to determine the statistical predictions of that particular inflationary model. Since the simulation of any one Gaussian random potential and subsequent field evolution is completely independent from the next, I implemented methods to parallelize simulations. These methods originate largely from the `concurrent.futures` module in Python, allowing independent simulations to run on different cores on a computer simultaneously. The University of Richmond supercomputing cluster is optimized to handle parallelized code, boasting hundreds of cores spread throughout dozens of nodes. Code parallelization, when paired with the cluster, improved the efficiency of my program by an order of magnitude.

VI. RESULTS AND PREDICTIONS

The results of this project will be presented in the form of statistical predictions of perturbation spectra made by a Gaussian random potential with three sets of fixed parameters. I chose these parameters through a mixture of theory and trial-and-error to exemplify the promise of my model.

$$\begin{aligned} s &= 30 \\ V_0^2 &= 5 \cdot 10^{-9} \\ N &\in \{1, 2, 3\} \end{aligned}$$

The three sets of parameters are, thus $(30, 5 \cdot 10^{-9}, 1), (30, 5 \cdot 10^{-9}, 2)$, and $(30, 5 \cdot 10^{-9}, 3)$. The parameter space to explore within these three degrees of freedom s, V_0^2, N is huge, leaving plenty of room for further analysis.

A. The Importance of Repeated Trials

With any fixed parameter set, I need to gather a large-enough sample of data to represent *all possible successful trajectories* under those generations. Success, phrased in the language of Section IV F, means *not* failing. The trajectory has to:

1. converge to the origin, and
2. accumulate enough e-folds of inflation along the way.

To compute a representative sample of successful trajectories, and subsequently their predicted observable quantities, I need to vary my initial starting condition $\boldsymbol{\phi}_0$. As previously mentioned, the only relevant quantity related to this initial condition is its distance from the minimum at the origin, $|\boldsymbol{\phi}_0| = \phi_0$. There are successful trajectories evolved under potentials generated by my parameter set that start at $\phi_0 = s/4$ all the way out to $\phi_0 = 2s$. Here is my method to handling this variety within a given parameter set:

- To account for all of these trajectories, I subdivide $\boldsymbol{\phi}$-space into N-dimensional shells centered around the origin with thicknesses of $s/10$ each. The first shell measures from $\phi_0 = 0.3s$ to $0.4s$, and the last from $\phi_0 = 2.0s$ to $2.1s$.
- Within each shell, I conduct 1000 simulations. That is, I randomly choose $\boldsymbol{\phi}_0$ such that its magnitude falls within the first shell, develop a random trajectory, and compute its predictions for perturbation spectra if possible, all 1000 times. Then, I move on to the next shell farther out and do the same.
- Once I have computed and stored the trajectories and predictions within each shell, I move on to the next parameter set and repeat the whole process.

By the time the University of Richmond supercomputing cluster outputs all of the simulated information, it will have computed at least 36000 trajectories. Currently, these jobs take around 22 hours when submitted to one node of the cluster. Once the jobs are complete, it is time to analyze the data properly.

B. Success Probabilities

To calculate the statistical prediction of, say, A_S from a given set of parameters, it is not mathematically sound to take the simple mean and standard deviation of the A_S values associated with the entire bunch of trajectories. It is necessary to account for the fact that there is not an equal likelihood that a trajectory with a given ϕ_0 is successful. Trajectories with low ϕ_0 (starting close to the minimum at the origin) have a very high chance of converging, but a low chance of accumulating enough e-folds of inflation. On the other hand, trajectories with high ϕ_0 (starting far from the origin) are unlikely to converge to the origin, but if they do, they almost always accumulate a surplus of inflation. A meaningful prediction of A_S and other quantities comes by weighting the mean and standard deviation appropriately:

- For each shell, there is an associated probability of success $P(S)$. The probability is easily calculable by taking the ratio of the number of simulations that converged and accumulated enough inflation to the total number 1000.
- Since shells correspond with ranges for ϕ_0, $P(S)$ can be considered a function of radial distance. At its extremes, $P(S) \downarrow 0$ as $\phi_0 \downarrow 0$ and $P(S) \downarrow 0$ as $\phi_0 \uparrow \infty$. Interestingly, $P(S)$ should have a maximum at some $\phi_0 > 0$.
- The weighted mean of some quantity q is then

$$\mu_q = \frac{\sum_l w_l q_l}{\sum_l w_l} \tag{59}$$

  where w_l is the $P(S)$ in a particular shell l and q_l is the simple mean of the q within l. The appropriately weighted variance is

$$\sigma_q^2 = \frac{\sum_l w_l q_l^2}{\sum_l w_l} - \mu_q^2 \tag{60}$$

For our sets of parameters, we can visualize the probability of success as a function of ϕ_0. Analyzed from a supercomputer output, the curves are shown in Figure 9. Theoretically, this curve is likely the product of two probability distributions: the probability of convergence to the origin $P(C)$ and the probability of attaining enough inflation $P(I)$. As the probability of convergence to the origin rapidly decreases as ϕ_0 grows, $P(C)$ is likely modeled by exponential decay. $P(I)$ most likely follows a power law in ϕ_0, proportional to the volume in $\boldsymbol{\phi}$-space enclosed at by a sphere of radius ϕ_0.

C. Perturbation Spectra

Finally, predictions for the perturbation spectra are presented in this section. First, we can examine weighted

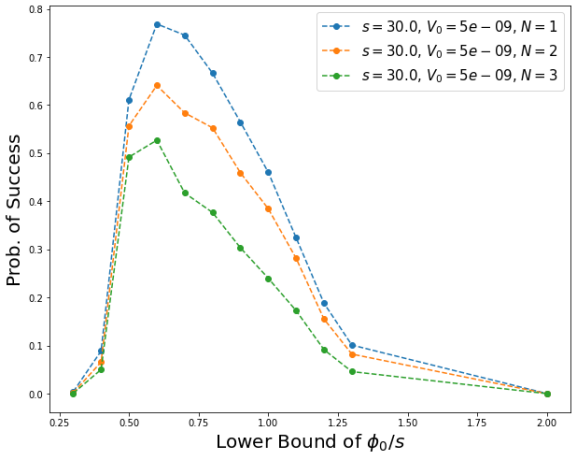

FIG. 9. The probability that a trajectory starting at an initial distance ϕ_0 from the origin will both converge and accumulate enough inflation. I did not simulate in the shells between $1.4s$ and $2.0s$ to save time, since the behavior would be well documented by a single data point at $2.0s$.

histograms detailing the value of some quantity q by the frequency of its prediction, weighted by the general successfulness of trajectories in each shell in Figure 10. Each histogram corresponds to predictions for a particular quantity under a particular set of parameters. For instance, the second column of plots corresponds to the parameter set $(s, V_0^2, N) = (30, 5 \cdot 10^{-9}, 2)$. Five histograms for five different observable quantities related to perturbation spectra (A_S, n_S, A_T, n_T, and r) are presented separately in each row. Notice a few things:

- Predictions for each observable quantity appear to be distributed near-normally, so standard deviations of these quantities are an appropriate statistic to measure.
- Different ϕ_0 shells (colors) sometimes yield different distributions for predictions. Individually, the shells may predict different means and standard deviations of each quantity, but the statistics for individual shells are not as important as overall statistics for the set of parameters (the weighted sum of all shells).
  - A good example of this phenomenon is clearly visible in the n_T histograms. Shells close around the origin (darker blues) tend predict higher values for n_T than shells farther from the origin (warmer colors). The secondary peaks near $n_T = 0$ are very curious!
- The horizontal scales are the same across each row.
- There are predicted values outside the horizontal plot range shown, but the ranges shown encompass the vast majority of the information.

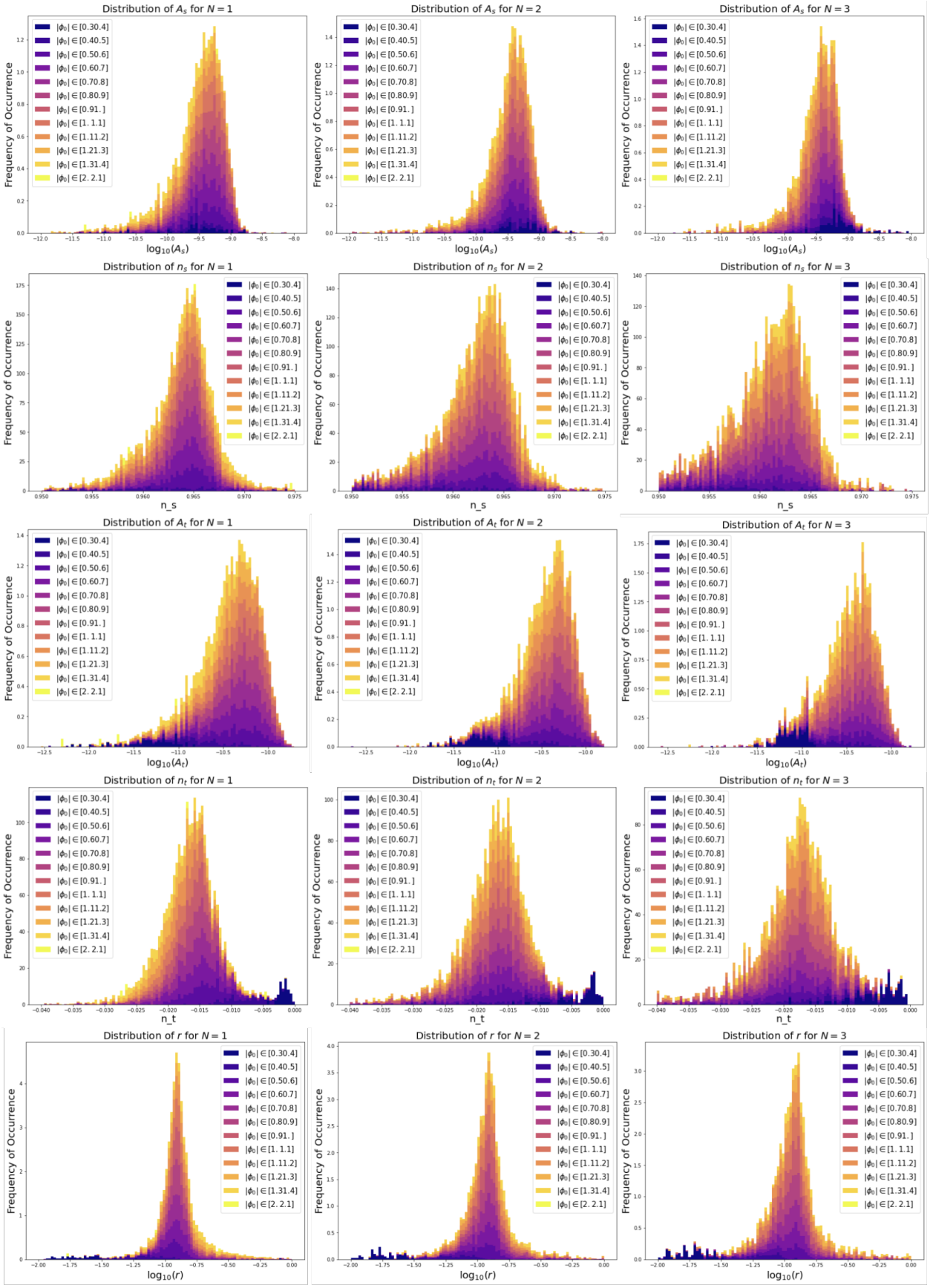

FIG. 10. Weighted histograms of observable quantities A_S, n_S, A_T, n_T, and r. Different colors on each histogram correspond to different initial positions ϕ_0. Each column of plots corresponds to a different set of parameters (in this case, just changing the number of dimensions N) which defined how the Gaussian random potential was generated.

Now that we have a good way to visual the weighted predictions in each parameter set, we can introduce the numerical predictions for quantities in the form $q = \mu_q \pm \sigma_q$. For $N = 1$,

$$\begin{aligned} \log_{10}(A_S) &= -9.4727 \pm 0.0909 \qquad (-8.680) \\ n_S &= 0.96360 \pm 0.00205 \qquad (0.9626) \\ \log_{10}(A_T) &= -10.404 \pm 0.118 \\ n_T &= -0.01600 \pm 0.00218 \\ \log_{10}(r) &= -0.90242 \pm 0.07248 \qquad (< -1.0) \end{aligned}$$

where the values in parentheses are the mean predictions by Planck data, if available, or the general constraint for r. For $N = 2$,

$$\begin{aligned} \log_{10}(A_S) &= -9.4875 \pm 0.2465 \\ n_S &= 0.96024 \pm 0.02427 \\ \log_{10}(A_T) &= -10.440 \pm 0.073 \\ n_T &= -0.01781 \pm 0.00101 \\ \log_{10}(r) &= -0.92279 \pm 0.06822 \end{aligned}$$

Finally, for $N = 3$,

$$\begin{aligned} \log_{10}(A_S) &= -9.5013 \pm 0.2834 \\ n_S &= 0.95786 \pm 0.02705 \\ \log_{10}(A_T) &= -10.453 \pm 0.089 \\ n_T &= -0.01909 \pm 0.00095 \\ \log_{10}(r) &= -0.92139 \pm 0.07632 \end{aligned}$$

These predictions are very promising. Values for the spectral index n_S are spot-on; the accepted value lies within one standard deviation for all $N = 1, 2, 3$. Quantities A_S and r are not far off. These predictions come from only three sets of parameters out of a huge volume of parameter space. Given the proximity of predicted values to the accepted values, it seems that further effort in tweaking s, V_0^2, and N could certainly close the gaps between predictions and observations.

There is one more productive way to visualize the perturbation spectra predictions by my model. I can place one quantity q_1 on a horizontal axis, another q_2 on a vertical axis, and scatter plot points (q_1, q_2) for each trajectory. The result looks like a slightly skewed cloud of points centered on (μ_{q_1}, μ_{q_2}). Mathematically, they will appear to be points randomly sampled from a multivariate Normal distribution. With this data we can perform a *Gaussian kernel density estimation* to describe the density of the plotted points at each point in the space of the quantities. This is done for a few quantities in Figure 11. Notice a few more things about this representation:

- The scatter plots are on the left side. Each point represents an individual trajectory, with coloring exactly the same as in Figure 10. As with the histograms, there are some data points not encompassed within the chosen domains. In fact, the data presented in these scatter plots is exactly the same as the data presented in their corresponding histograms.
- The kernel density estimations are on the right. The lighter colors correspond to the regions on the scatter plot most densely populated with points. The closed black curves are analogous to lines of one and two standard deviations from the weighted mean.
- The kernel density estimations look more compact than the scatter plots, but that is only because there is such a high density of points in the corresponding regions of the scatter plots.
- Approximately 11000 points are plotted on each of the scatter plots, plotted on top of each other in order from low ϕ_0 (blues) to high ϕ_0 (oranges). This plotting method effectively shows some of all the colors, but hides some of the blue points.

Kernel density estimations could be applied to every distinct pair of perturbation spectra quantities, an effective way to represent the high-density data.

VII. CONCLUSIONS

The N-dimensional Gaussian random potential is a highly promising inflationary model that is hefty to test computationally, but responsive to data compression and optimization methods. The potential function does not naturally spawn the minimum at $V = 0$ customary in most inflationary models, but the minimum can be seeded by imposing the appropriate conditions. The covariance matrix of constraints on V is the source of both triumphs and problems in this project. While it effectively simulates data about potential functions, including derivatives, it can quickly become unwieldy and ill-conditioned. Data compression and efficiency algorithms are necessary to generate the requisite amount of information to accurately produce statistical predictions, especially in higher N. Preliminary results suggest that some regions of (s, V_0, N) parameter-space produce potentials that evolve their corresponding universes in ways that nearly match observed properties of our own universe. Other inflationary potentials have successfully replicated these same properties, but characterization of the Gaussian random model to the extent possible with data compression algorithms is unique to this project.

A. Future Work

There are quite a few tasks left to extend this project to its full potential. Some major steps going forward are enumerated below.

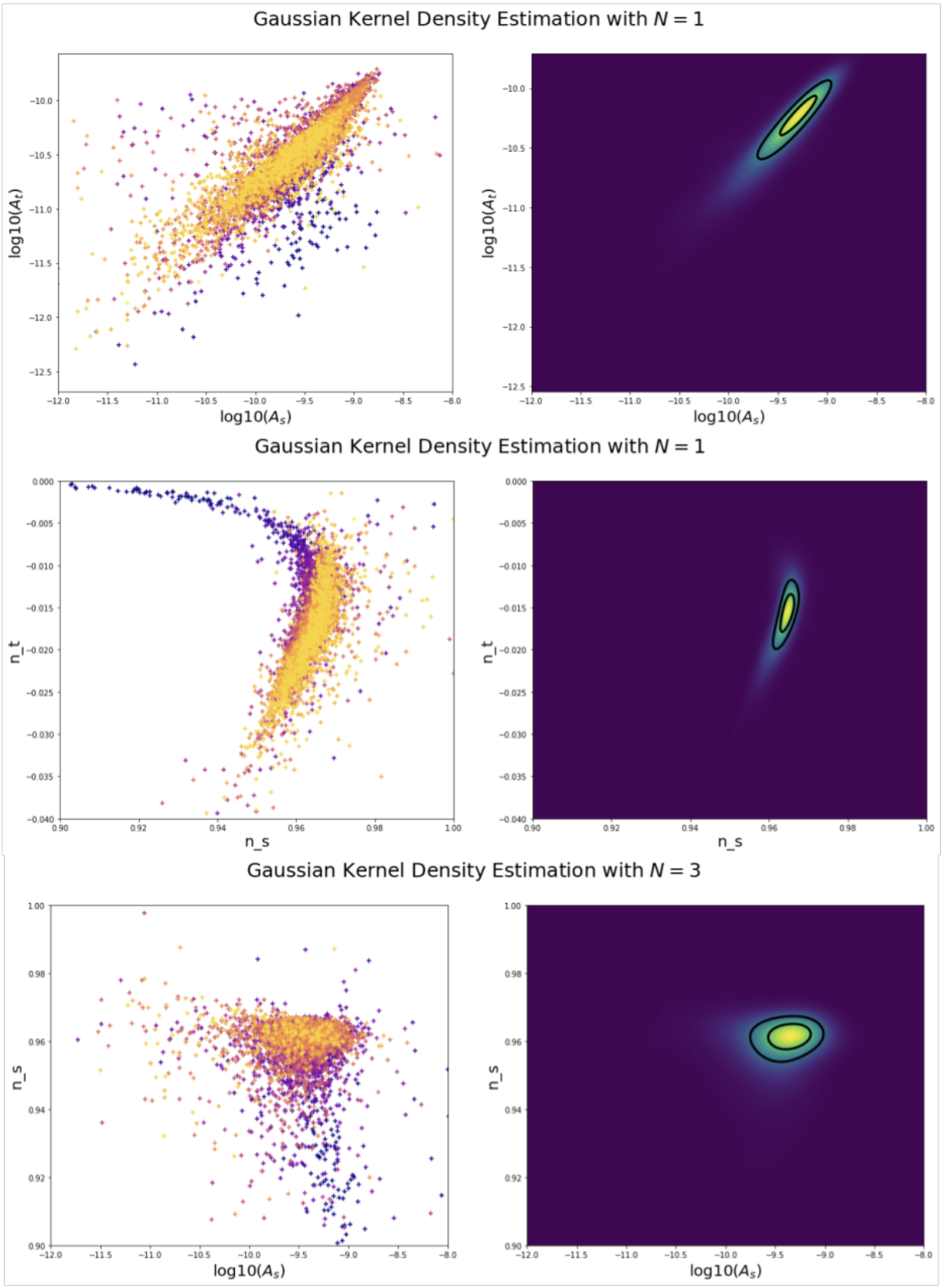

FIG. 11. Gaussian kernel density estimations (KDEs) of scatter-plotted predictions for perturbation spectra. Each cross on the scatter plots corresponds to the prediction of one individual trajectory. The colors of crosses are identical to those in Figure 10. About 11000 individual points are plotted, but many are obscured by points plotted slightly later. The KDEs express the density of the points on the scatter plot. The closed black curves indicate the areas enclosing 68% (inner) and 95% (outer) of the weighted distribution of data.

1. *Implement principled forgetting efficiently.* Algorithmic data compression described in Section V F is promising in its ability to effectively compress the covariance matrix of constraints on V, but left unimplemented due to computational struggles. Resolving these struggles is likely the path to accumulating large quantities of data about Gaussian potentials with high numbers of dimensions.

2. *Explore parameter space.* A very restricted region of parameter space has been explored thus far. Continuing to utilize the supercomputing capabilities of the University of Richmond cluster, we can continue to tinker with s, V_0, and N in hopes of closing the gap between predictions and observed constraints.

3. *Translate to other languages.* Python is a good choice programming language for alpha testing, but it is not the most efficient language for scientific computing. Translating the code to languages such as C or FORTRAN are necessary to fully maximize computational efficiency.

4. *Incorporate other observable quantities.* Some inflationary models have been tested against a handful of other observable quantities, such as non-Gaussianity of the CMB [7], to provide more rigorous tests of plausibility. Working out the theory and incorporating these additional tests could "make or break" the Gaussian random model; they could either provide additional depth of certainty in the plausibility of the model or seed doubt in its predictions.

VIII. ACKNOWLEDGMENTS

This thesis was supported by two Summer Research Fellowships from the University of Richmond School of Arts & Sciences from May 2019 to July 2020. My results were only obtainable in such great quantity thanks to the University of Richmond supercomputing cluster. A special thanks to the Physics Department in their unwavering support. I am indebted to Dr. Ted Bunn for the brilliant insight, relentless patience, and inspiring creativity that characterizes this project to its core.

IX. APPENDIX

A. Deriving $\mathcal{P}_S(\boldsymbol{k})$ From the Trajectory

The *mode matrix* $\boldsymbol{\Psi}$ is an $N \times N$ matrix of functions of time that obeys the following differential equation, element-wise.

$$\frac{d^2\Psi^{(IJ)}}{dN_e^2} + (1-\epsilon)\frac{d\Psi^{(IJ)}}{dN_e} + \left(\frac{k^2}{a^2H^2} - 2 + \epsilon\right)\Psi^{(IJ)} + C^{(IL)}\Psi^{(LJ)} = 0 \quad (61)$$

where Einstein summation convention is employed on the last term and $C^{(IJ)}$ is the *coupling tensor*,

$$C^{(IJ)} = \frac{\partial^2 V}{\partial\phi^{(I)}\partial\phi^{(J)}} + \frac{1}{H^2}\left(\frac{d\phi^{(I)}}{dN_e}\frac{\partial V}{\partial\phi^{(J)}} + \frac{d\phi^{(J)}}{dN_e}\frac{\partial V}{\partial\phi^{(I)}}\right) + (3-\epsilon)\frac{d\phi^{(I)}}{dN_e}\frac{d\phi^{(J)}}{dN_e}. \quad (62)$$

This equation is the lone place in this project in which the second derivatives of the potential are necessary. The differential equation (61) can be solved for the mode matrix with initial conditions

$$\Psi^{(IJ)}(N_e = 0) = \frac{1}{\sqrt{2k(0)}}\delta^{(IJ)} \quad (63)$$

$$\frac{d\Psi^{(IJ)}}{dN_e}(N_e = 0) = -\frac{i}{a(0)H(0)}\sqrt{\frac{k(0)}{2}}\delta^{(IJ)} \quad (64)$$

where $\delta^{(IJ)}$ is the Kronecker delta. Then, define the $\delta\phi$ power spectrum matrix as

$$\mathcal{P}_{\delta\phi}^{(IJ)}(k) = \frac{k^3}{2\pi^2}\frac{1}{a^2}\Psi^{(IL)}(\Psi^*)^{(LJ)} \quad (65)$$

where $\boldsymbol{A}^*$ is the Hermitian adjoint of $\boldsymbol{A}$. Now, the power spectrum of scalar perturbations is the 00 term of the following matrix, expressable in terms of the $\delta\phi$ power spectrum matrix.

$$\mathcal{P}_S^{(IJ)}(k) = \frac{1}{2\epsilon}\omega_I\omega_J\mathcal{P}_{\delta\phi}^{(IJ)}(k) \quad (66)$$

In this case, $\omega_I = \dot{\phi}^{(I)}/|\dot{\boldsymbol{\phi}}|$ and functions a and ϵ are functions of k. To be clear, $\mathcal{P}_S(k) = \mathcal{P}_S^{00}(k)$.

B. Inputs and Outputs of `bushwhack`

The most important function that I coded for this project is labeled `bushwhack`. The user has many *input parameters* to change that change the simulation of the trajectory, the extraction of observable quantities from the trajectory, or the computational process underlying. They are listed with their variable names, default values, and described purposes in Table I.

The `bushwhack` function outputs an object from a completely original class `BStuff`. `BStuff` objects are designed to hold all the necessary information about the trajectory, predicted observable quantities, and the computational process involved in the corresponding run. The structure and design of this object is critical for efficient manual analysis and code development. Find the attributes of `BStuff` objects described in Table II.

Parameter Name	Default Value	Purpose
`s`	30	Value for the coherence scale of the potential.
`V0`	5e-9	Value for the inflationary energy scale of the potential.
`N`	2	Value for the number of inflaton fields, i.e. the number of dimensions of V.
`from0_bounds`	$(0.2, 0.8)$	Bounds on the starting distance from the origin. ϕ_0 will be randomly generated from a uniform distribution between these two values.
`nE_bounds`	$(45, 500)$	Bounds on the acceptable predicted number of e-folds of inflation. If the program predicts (as in Section V A) a value for total N_e that is outside these bounds, the program restarts.
`psi_integrate`	$(-3, \texttt{None})$	Parameter for the integration bounds when finding a solution of the differential equation (61).
`fails_coef`	100	The number of times a run is allowed to predict a failure before completely shutting down. Scaled by 2^N since there is a higher chance for failure in higher dimensions.
`maxJumpFrac`	1/20	The maximum fraction of the coherence length that the differential equation solver can jump in one time step.
`nE_buffer`	2	Another parameter for solving equation (61).
`state`	`None`	If entered, reverts the computer back to a previous random state and successfully replicates a previously generated random potential. Powerful parameter for analysis.
`method`	`'RK45'`	The numerical solution method for the differential equations of the trajectory. Passed immediately into `solve_ivp`. Defaults to fourth-order Runge-Kutta.
`name`	`None`	Adds a completely unnecessary attribute to the resultant `BStuff` object so that it is callable by a different name other than its variable name.

TABLE I. All input parameters into the `bushwhack` function.

REFERENCES

[1] Y. Akrami et al. "Planck2018 results". In: *Astronomy Astrophysics* 641 (Sept. 2020), A10. ISSN: 1432-0746. DOI: 10.1051/0004-6361/201833887. URL: http://dx.doi.org/10.1051/0004-6361/201833887.

[2] Scott Dodelson. *Modern Cosmology*". 1st ed. Academic Press, 2003. ISBN: 9780122191411.

[3] D. Langlois. "Inflation and Cosmological Perturbations". In: *Lecture Notes in Physics* (2010), pp. 1–57. ISSN: 1616-6361. DOI: 10.1007/978-3-642-10598-2_1. URL: http://dx.doi.org/10.1007/978-3-642-10598-2_1.

[4] William H. Press et al. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. 3rd ed. USA: Cambridge University Press, 2007. ISBN: 0521880688.

[5] Layne C. Price et al. "Designing and testing inflationary models with Bayesian networks". In: *Journal of Cosmology and Astroparticle Physics* 2016.02 (Feb. 2016), pp. 049–049. DOI: 10.1088/1475-7516/2016/02/049. URL: https://doi.org/10.1088/1475-7516/2016/02/049.

[6] Layne C. Price et al. "MULTIMODECODE: an efficient numerical solver for multifield inflation". In: *Journal of Cosmology and Astroparticle Physics* 2015.03 (Mar. 2015), pp. 005–005. ISSN: 1475-7516. DOI: 10.1088/1475-7516/2015/03/005. URL: http://dx.doi.org/10.1088/1475-7516/2015/03/005.

[7] Sébastien Renaux-Petel. "Combined local and equilateral non-Gaussianities from multifield DBI inflation". In: *Journal of Cosmology and Astroparticle Physics* 2009.10 (Oct. 2009), pp. 012–012. ISSN: 1475-7516. DOI: 10.1088/1475-7516/2009/10/012. URL: http://dx.doi.org/10.1088/1475-7516/2009/10/012.

[8] Barbara Ryden. *Introduction to Cosmology*. 2nd ed. Cambridge University Press, 2002. ISBN: 9781316889848.

[9] S.-H. Henry Tye, Jiajun Xu, and Yang Zhang. "Multifield inflation with a random potential". In: *Journal of Cosmology and Astroparticle Physics* 2009.04 (Apr. 2009), pp. 018–018. ISSN: 1475-7516. DOI: 10.1088/1475-7516/2009/04/018. URL: http://dx.doi.org/10.1088/1475-7516/2009/04/018.

Attribute Name	**Description**
`phi`	Values of inflaton field $\boldsymbol{\phi}$ along the trajectory.
`dphi`	Values of $d\boldsymbol{\phi}/dN_e$ along the trajectory.
`V`	Values of the inflaton potential V along the trajectory.
`dV`	Values of $\nabla_\phi V$ along the trajectory.
`ddV`	Values of all the mixed second partial derivatives of V along the trajectory.
`t`	Values of cosmic time t along the trajectory.
`nE`	Values of the number of e-folds N_e along the trajectory.
`params`	A list of parameters on the potential: (s, V_0^2, N).
`H`	Values of the Hubble parameter along the trajectory.
`a`	Values of the scale factor along the trajectory.
`epsilon`	Values of the stopping criterion ϵ along the trajectory.
`k`	Values of the horizon-crossing modes k along the trajectory.
`inw`	Values of the inwardness ι along the trajectory.
`vcov`	The (potentially compressed) covariance matrix $\boldsymbol{\Gamma}$ of constraints on the potential.
`vcov_L`	The (potentially compressed) Cholesky decomposition of $\boldsymbol{\Gamma}$.
`beefCov`	The uncompressed covariance matrix containing all constraints on the potential.
`beefCov_L`	The uncompressed Cholesky decomposition of the covariance matrix.
`vFull`	The ordered list $\boldsymbol{v}$ of all potential information, including derivatives.
`bowl`	Information about the minimum constraints at the origin.
`state`	The random state of the computer at the time of the simulation.
`nfev`	The number of times the numerical differential equation solver was called.
`simtime`	The amount of time the simulation took to finish.
`success`	Details about if and how the trajectory was determined to be remotely plausible.
`forgets`	The number of times the code executed an "unprincipled forgetting" step as in Section V E.
`name`	Completely unnecessary name of the run; just for fun.
`coupler`	The coupling tensor $\boldsymbol{C}$ from equation (61).
`A_s`	The predicted scalar amplitude.
`n_s`	The predicted scalar spectral index.
`A_t`	The predicted tensor amplitude.
`n_t`	The predicted tensor spectral index.
`r`	The predicted tensor-to-scalar ratio.

TABLE II. All attributes of the outputs of the `bushwhack` function.