Fast and Scalable Gaussian Process Modeling with Applications to Astronomical Time Series

Daniel Foreman-Mackey[1,2,6] , Eric Agol[1,7] , Sivaram Ambikasaran[3] , and Ruth Angus[4,5]

[1] Astronomy Department, University of Washington, Seattle, WA, USA
[2] Center for Computational Astrophysics, Flatiron Institute, 162 5th Avenue, 6th Floor, New York, NY 10010, USA
[3] Department of Computational and Data Sciences, Indian Institute of Science, Bangalore, India
[4] Department of Astronomy, Columbia University, 550 W 120th Street, New York, NY 10027, USA

Abstract

The growing field of large-scale time domain astronomy requires methods for probabilistic data analysis that are computationally tractable, even with large data sets. Gaussian processes (GPs) are a popular class of models used for this purpose, but since the computational cost scales, in general, as the cube of the number of data points, their application has been limited to small data sets. In this paper, we present a novel method for GPs modeling in one dimension where the computational requirements scale linearly with the size of the data set. We demonstrate the method by applying it to simulated and real astronomical time series data sets. These demonstrations are examples of probabilistic inference of stellar rotation periods, asteroseismic oscillation spectra, and transiting planet parameters. The method exploits structure in the problem when the covariance function is expressed as a mixture of complex exponentials, without requiring evenly spaced observations or uniform noise. This form of covariance arises naturally when the process is a mixture of stochastically driven damped harmonic oscillators—providing a physical motivation for and interpretation of this choice—but we also demonstrate that it can be a useful effective model in some other cases. We present a mathematical description of the method and compare it to existing scalable GP methods. The method is fast and interpretable, with a range of potential applications within astronomical data analysis and beyond. We provide well-tested and documented open-source implementations of this method in C++, Python, and Julia.

Key words: asteroseismology – methods: data analysis – methods: statistical – planetary systems – stars: rotation

1. Introduction

In the astrophysical literature, Gaussian processes (GPs; Rasmussen & Williams 2006) have been used to model stochastic variability in light curves of stars (Brewer & Stello 2009), active galactic nuclei (Kelly et al. 2014), and the logarithmic flux of X-ray binaries (Uttley et al. 2005). They have also been used as models for the cosmic microwave background (Bond & Efstathiou 1987; Bond et al. 1999; Wandelt & Hansen 2003), correlated instrumental noise (Gibson et al. 2012), and spectroscopic calibration (Evans et al. 2015; Czekala et al. 2017). While these models are widely applicable, their use has been limited, in practice, by the computational cost and scaling. The cost of computing a general GP likelihood scales as the cube of the number of data points $\mathcal{O}(N^3)$, and in the current era of large time domain surveys—with as many as $\sim 10^4$–10^9 targets with $\sim 10^3$–10^5 observations each—this scaling is prohibitive.

Existing astronomical time series data sets have already reached the limit where naïve application of GP models is no longer tractable. NASA's Kepler mission (Borucki et al. 2010), for example, measured light curves with more than 60,000 observations each for about 190,000 stars. Current and forthcoming surveys such as K2 (Howell et al. 2014), TESS (Ricker et al. 2014), LSST (Ivezić et al. 2008), WFIRST (Spergel et al. 2015), and PLATO (Rauer et al. 2014) will continue to produce similar or larger data volumes.

In this paper, we present a method for directly and exactly computing a class of GP models that scales linearly with the number of data points $\mathcal{O}(N)$ for one-dimensional data sets. Unlike earlier linear methods using Kalman filters (e.g., Kelly et al. 2014), this novel algorithm exploits the semiseparable structure of a specific class of covariance matrices to directly factorize and solve the system. This method can only be used with one-dimensional data sets, and the covariance function must be represented by a mixture of exponentials; we will return to a discussion of what this means in detail in the following sections. However, the measurements do not need to be evenly spaced, and the uncertainties can be heteroscedastic.

This method achieves linear scaling by exploiting structure in the covariance matrix when it is generated by a mixture of exponentials. The semiseparable nature of these matrices was first recognized by Ambikasaran (2015), building on intuition from a 20 yr old paper (Rybicki & Press 1995). As we will discuss in the following pages, this choice of kernel function arises naturally in physical systems, and we demonstrate that it can be used as an effective model[8] in other cases. This method is especially appealing compared to other similar methods—we discuss these comparisons in Section 7—because it is exact, simple, and fast.

Our main expertise lies in the field of exoplanet discovery and characterization where GPs have become a model of choice. We are confident that this method will benefit this field, but we also expect that there will be applications in other branches of astrophysics and beyond. In Section 6, we present applications of the method to research problems in stellar rotation (Section 6.3),

---

[8] Here and throughout, we use the term "effective model" to refer to a model that we use to make inferences even though it is known to be wrong.

asteroseismic analysis (Section 6.4), and exoplanet transit fitting (Section 6.5). Some readers might consider starting with this section for motivating examples before delving into the detailed derivations of the earlier sections.

In the following pages, we motivate the general problem of GP modeling, derive our novel direct solver, and demonstrate the model's application on real and simulated data sets. Alongside this paper, we have released well-tested and documented open-source implementations written in C++, Python, and Julia. These are available online at GitHub (https://github.com/dfm/celerite) and Zenodo.

2. Gaussian Processes

GPs are stochastic models consisting of a mean function $\mu_\theta(\mathbf{x})$ and a covariance, autocorrelation, or "kernel" function $k_\alpha(\mathbf{x}_n, \mathbf{x}_m)$ parameterized by θ and α, respectively. Under this model, the log-likelihood function $\mathcal{L}(\theta, \alpha)$ with a data set

$$\mathbf{y} = (y_1 \quad \cdots \quad y_N)^T \qquad (1)$$

at coordinates

$$X = (\mathbf{x}_1 \quad \cdots \quad \mathbf{x}_N)^T \qquad (2)$$

is

$$
\begin{aligned}
\ln \mathcal{L}(\theta, \alpha) = \ln p(\mathbf{y} \mid X, \theta, \alpha) = \\
-\frac{1}{2}\mathbf{r}_\theta^T \, K_\alpha^{-1} \, \mathbf{r}_\theta - \frac{1}{2}\ln \det K_\alpha - \frac{N}{2}\ln(2\pi),
\end{aligned}
\qquad (3)
$$

where

$$\mathbf{r}_\theta = (y_1 - \mu_\theta(\mathbf{x}_1) \quad \cdots \quad y_N - \mu_\theta(\mathbf{x}_N))^T \qquad (4)$$

is the vector of residuals and the elements of the covariance matrix K are given by $[K_\alpha]_{nm} = k_\alpha(\mathbf{x}_n, \mathbf{x}_m)$. Equation (3) is the equation of an N-dimensional Gaussian, and it can be derived as the generalization of what we astronomers call "χ^2" to include the effects of correlated noise. A point estimate for the values of the parameters θ and α for a given data set $(\mathbf{y}, X)$ can be found by maximizing Equation (3)—or the likelihood times a prior for a Bayesian estimate—with respect to θ and α using a nonlinear optimization routine (Nocedal & Wright 2006). Furthermore, the posterior probability density for θ and α can be quantified by multiplying the likelihood by a prior $p(\theta, \alpha)$ and using a Markov chain Monte Carlo (MCMC) algorithm to sample the joint posterior probability density.

GP models have been widely used across the physical sciences, but their application is generally limited to small data sets, $N \lesssim 10^3$, because the cost of computing the inverse and determinant of a general matrix K_α is $\mathcal{O}(N^3)$. In other words, this cost is proportional to the cube of the number of data points N. Furthermore, the storage requirements also scale as $\mathcal{O}(N^2)$. This means that for large data sets every evaluation of the likelihood quickly becomes computationally intractable, and representing the matrix in memory can become expensive. In this case, the use of nonlinear optimization or MCMC is no longer practical.

This is not a purely academic concern. While the cost of directly evaluating the GP likelihood using a tuned linear algebra library[9]

with a data set of $N = 1024$ measurements is less than $1/10$ of a second, the same calculation with $N = 8192$ takes over 8 s. Furthermore, most optimization and sampling routines are iterative, requiring many evaluations of this model to perform parameter estimation or inference. Existing data sets from astronomical surveys like Kepler and K2 include tens of thousands of observations for hundreds of thousands of targets, and upcoming surveys like LSST are expected to produce thousands or tens of thousands of measurements each for billions of astronomical sources (Ivezić et al. 2008). Scalable methods will be required if we ever hope to use GP models with these data sets.

In this paper, we present a method for improving the cubic scaling for some use cases. We call our method and its implementations celerite.[10] The celerite method requires using a specific model for the covariance $k_\alpha(\mathbf{x}_n, \mathbf{x}_m)$, and although it has some limitations, we demonstrate in subsequent sections that it can increase the computational efficiency of many astronomical data analysis problems. The main limitation of this method is that it can only be applied to one-dimensional data sets, where by "one-dimensional" we mean that the *input coordinates* $\mathbf{x}_n$ are scalar, $\mathbf{x}_n \equiv t_n$. We use t as the input coordinate because one-dimensional GPs are often applied to time series data, but this is not a real restriction and the celerite method can be applied to *any* one-dimensional data set. Furthermore, the covariance function for the celerite method is "stationary." In other words, $k_\alpha(t_n, t_m)$ is only a function of $\tau_{nm} \equiv |t_n - t_m|$.

3. The Celerite Model

To scale GP models to larger data sets, Rybicki & Press (1995) presented a method of computing the first term in Equation (3) in $\mathcal{O}(N)$ operations when the covariance function is given by

$$k_\alpha(\tau_{nm}) = \sigma_n^2 \, \delta_{nm} + a \exp(-c \, \tau_{nm}), \qquad (5)$$

where $\{\sigma_n^2\}_{n=1}^N$ are the measurement uncertainties, δ_{nm} is the Kronecker delta, and $\alpha = (a, c)$. The intuition behind this method is that, for this choice of k_α, the inverse of K_α is tridiagonal and can be computed with a small number of operations for each data point. This model has been generalized to arbitrary mixtures of exponentials (e.g., Kelly et al. 2011)

$$k_\alpha(\tau_{nm}) = \sigma_n^2 \, \delta_{nm} + \sum_{j=1}^{J} a_j \exp(-c_j \, \tau_{nm}). \qquad (6)$$

In this case, the inverse is dense, but Equation (3) can still be evaluated with a scaling of $\mathcal{O}(N \, J^2)$, where J is the number of components in the mixture (Kelly et al. 2014; Ambikasaran 2015). In Section 5, we build on previous work (Ambikasaran 2015) to derive a faster method with the same $\mathcal{O}(N \, J^2)$ scaling that can be used when $k_\alpha(\tau_{nm})$ is positive definite.

This kernel function can be made even more general by introducing complex parameters $a_j \to a_j \pm i \, b_j$ and

---

[9] For comparisons throughout this paper, we use the Intel Math Kernel Library (https://software.intel.com/en-us/intel-mkl) running on a recent Apple MacBook Pro.

[10] The name "celerite" comes from the French word *célérité* meaning the speed of light in a vacuum. Throughout the paper, when referring to the *celerite* model, we typeset *celerite* in italics, and in reference to the implementation of this model, we typeset celerite in roman font.

$c_j \to c_j \pm i\, d_j$. In this case, the covariance function becomes

$$
k_\alpha(\tau_{nm}) = \sigma_n^2\, \delta_{nm} + \sum_{j=1}^{J}
$$
$$
\times \left[\frac{1}{2}(a_j + i\, b_j) \exp(-(c_j + i\, d_j)\tau_{nm}) \right.
$$
$$
\left. + \frac{1}{2}(a_j - i\, b_j) \exp(-(c_j - i\, d_j)\tau_{nm}) \right], \quad (7)
$$

and, for this function, Equation (3) can still be evaluated with $\mathcal{O}(N\, J^2)$ operations. The details of this method and a few implementation considerations are outlined in Section 5, but we first discuss some properties of this covariance function.

By rewriting the exponentials in Equation (7) as sums of sine and cosine functions, we can see that the autocorrelation structure is defined by a mixture of quasi-periodic oscillators

$$
k_\alpha(\tau_{nm}) = \sigma_n^2\, \delta_{nm} + \sum_{j=1}^{J}
$$
$$
\times [a_j \exp(-c_j\, \tau_{nm}) \cos(d_j\, \tau_{nm})
$$
$$
+ b_j \exp(-c_j\, \tau_{nm}) \sin(d_j\, \tau_{nm})]. \quad (8)
$$

For clarity, we refer to the argument within the sum as a "*celerite* term" for the remainder of this paper. The Fourier transform[11] of this covariance function is the power spectral density (PSD) of the process, and it is given by

$$
S(\omega) = \sum_{j=1}^{J} \sqrt{\frac{2}{\pi}} \frac{(a_j\, c_j + b_j\, d_j)(c_j^2 + d_j^2) + (a_j\, c_j - b_j\, d_j)\omega^2}{\omega^4 + 2(c_j^2 - d_j^2)\omega^2 + (c_j^2 + d_j^2)^2}. \quad (9)
$$

The physical interpretation of this model is not immediately obvious, and we return to a more general discussion shortly, but we start by considering some special cases.

If we set the imaginary amplitude b_j for some component j to zero, that term of Equation (8) becomes

$$
k_j(\tau_{nm}) = a_j \exp(-c_j\, \tau_{nm}) \cos(d_j\, \tau_{nm}), \quad (10)
$$

and the PSD for this component is

$$
S_j(\omega) = \frac{1}{\sqrt{2\pi}} \frac{a_j}{c_j} \left[\frac{1}{1 + \left(\frac{\omega - d_j}{c_j}\right)^2} + \frac{1}{1 + \left(\frac{\omega + d_j}{c_j}\right)^2} \right]. \quad (11)
$$

This is the sum of two Lorentzian or Cauchy distributions with width c_j centered on $\omega = \pm d_j$. This model can be interpreted intuitively as a quasi-periodic oscillator with amplitude $A_j = a_j$, quality factor $Q_j = d_j\, (2\, c_j)^{-1}$, and period $P_j = 2\pi\, d_j^{-1}$.

Similarly, setting both b_j and d_j to zero, we get

$$
k_j(\tau_{nm}) = a_j \exp(-c_j\, \tau_{nm}), \quad (12)
$$

with the PSD

$$
S_j(\omega) = \sqrt{\frac{2}{\pi}} \frac{a_j}{c_j} \frac{1}{1 + \left(\frac{\omega}{c_j}\right)^2}. \quad (13)
$$

---

[11] Here and throughout we have defined the Fourier transform of the function $f(t)$ as $F(\omega) = (2\pi)^{-1/2} \int_{-\infty}^{\infty} f(t) e^{i\,\omega\, t}\, dt$.

This model is often called a "damped random walk" or Ornstein–Uhlenbeck process (in reference to the classic paper, Uhlenbeck & Ornstein 1930), and this is the kernel that was studied by Rybicki & Press (1992, 1995).

Finally, we note that the product of two terms of the form found inside the sum in Equation (8) can also be rewritten as a sum with updated parameters

$$
k_j(\tau)\, k_i(\tau) = e^{-\tilde{c}\, \tau} \left[\tilde{a}_+ \cos(\tilde{d}_+\, \tau) \right.
$$
$$
\left. + \tilde{b}_+ \sin(\tilde{d}_+\, \tau) + \tilde{a}_- \cos(\tilde{d}_-\, \tau) + \tilde{b}_- \sin(\tilde{d}_-\, \tau) \right], \quad (14)
$$

where

$$
\tilde{a}_\pm = \frac{1}{2}(a_j\, a_i \pm b_j\, b_i) \quad (15)
$$

$$
\tilde{b}_\pm = \frac{1}{2}(b_j\, a_i \mp a_j\, b_i) \quad (16)
$$

$$
\tilde{c} = c_j + c_i \quad (17)
$$

$$
\tilde{d}_\pm = d_j \mp d_i. \quad (18)
$$

Therefore, the method described in Section 5 can be used to perform scalable inference on large data sets for any model, where the kernel function is a sum or product of *celerite* terms.

4. Celerite as a Model of Stellar Variations

We now turn to a discussion of *celerite* as a model of astrophysical variability. A common concern in the context of GP modeling in astrophysics is the lack of physical motivation for the choice of kernel functions. Kernels are often chosen simply because they are popular, with little consideration of the impact of this decision. In this section, we discuss an exact physical interpretation of the *celerite* kernel that is applicable to many astrophysical systems, but especially the time variability of stars.

Many astronomical objects are variable on timescales determined by their physical structure. For phenomena such as stellar (asteroseismic) oscillations, variability is excited by noisy physical processes and grows most strongly at the characteristic timescale but is also damped owing to dissipation in the system. These oscillations are strong at resonant frequencies determined by the internal stellar structure, which are both excited and damped by convective turbulence.

To relate this physical picture to *celerite*, we consider the dynamics of a stochastically driven damped simple harmonic oscillator (SHO). The differential equation for this system is

$$
\left[\frac{d^2}{dt^2} + \frac{\omega_0}{Q} \frac{d}{dt} + \omega_0^2 \right] y(t) = \epsilon(t), \quad (19)
$$

where ω_0 is the frequency of the undamped oscillator, Q is the quality factor of the oscillator, and $\epsilon(t)$ is a stochastic driving force. If $\epsilon(t)$ is white noise, the PSD of this process is (Anderson et al. 1990)

$$
S(\omega) = \sqrt{\frac{2}{\pi}} \frac{S_0\, \omega_0^4}{(\omega^2 - \omega_0^2)^2 + \omega_0^2 \omega^2/Q^2}, \quad (20)
$$

where S_0 is proportional to the power at $\omega = \omega_0$, $S(\omega_0) = \sqrt{2/\pi}\, S_0\, Q^2$. The power spectrum in Equation (20) matches

Equation (9) if

$$a_j = S_0 \, \omega_0 \, Q$$

$$b_j = \frac{S_0 \, \omega_0 \, Q}{\sqrt{4 \, Q^2 - 1}}$$

$$c_j = \frac{\omega_0}{2 \, Q}$$

$$d_j = \frac{\omega_0}{2 \, Q} \sqrt{4 \, Q^2 - 1} \,, \tag{21}$$

for $Q \geqslant \frac{1}{2}$. For $0 < Q \leqslant \frac{1}{2}$, Equation (20) can be captured by a pair of *celerite* terms with parameters

$$a_{j\pm} = \frac{1}{2} \, S_0 \, \omega_0 \, Q \left[1 \pm \frac{1}{\sqrt{1 - 4 \, Q^2}} \right]$$

$$b_{j\pm} = 0$$

$$c_{j\pm} = \frac{\omega_0}{2 \, Q} [1 \mp \sqrt{1 - 4 \, Q^2}]$$

$$d_{j\pm} = 0. \tag{22}$$

For these definitions, the kernel is

$$k_{\text{SHO}}(\tau; S_0, Q, \omega_0) = S_0 \, \omega_0 \, Q \, e^{-\frac{\omega_0 \tau}{2Q}}$$

$$\begin{cases} \cosh(\eta \, \omega_0 \, \tau) + \dfrac{1}{2 \, \eta \, Q} \sinh(\eta \, \omega_0 \, \tau), & 0 < Q < 1/2 \\ 2(1 + \omega_0 \, \tau), & Q = 1/2 \\ \cos(\eta \, \omega_0 \, \tau) + \dfrac{1}{2 \, \eta \, Q} \sin(\eta \, \omega_0 \, \tau), & 1/2 < Q \end{cases},$$

$$\tag{23}$$

where $\eta = |1 - (4 \, Q^2)^{-1}|^{1/2}$. Because of the damping, the characteristic oscillation frequency in this model, d_j, for any finite quality factor $Q > 1/2$, is not equal to the frequency of the undamped oscillator, ω_0.

The power spectrum in Equation (20) has several limits of physical interest:

1. For $Q = 1/\sqrt{2}$, Equation (20) simplifies to

$$S(\omega) = \sqrt{\frac{2}{\pi}} \, \frac{S_0}{(\omega/\omega_0)^4 + 1}. \tag{24}$$

This functional form is commonly used to model for the background granulation noise in asteroseismic and helioseismic analyses (Harvey 1985; Michel et al. 2009; Kallinger et al. 2014). The kernel for this value of Q is

$$k(\tau) = S_0 \, \omega_0 \, e^{-\frac{1}{\sqrt{2}} \omega_0 \tau} \cos\left(\frac{\omega_0 \tau}{\sqrt{2}} - \frac{\pi}{4} \right). \tag{25}$$

2. Substituting $Q = 1/2$, Equation (20) becomes

$$S(\omega) = \sqrt{\frac{2}{\pi}} \, \frac{S_0}{[(\omega/\omega_0)^2 + 1]^2}, \tag{26}$$

with the corresponding covariance function (using Equations (8) and (22))

$$k(\tau) = \lim_{f \to 0} \frac{1}{2} \, S_0 \, \omega_0 [(1 + 1/f) e^{-\omega_0 (1-f)\tau}$$

$$+ (1 - 1/f) e^{-\omega_0 (1+f)\tau}] \tag{27}$$

$$= S_0 \, \omega_0 \, e^{-\omega_0 \tau} \, [1 + \omega_0 \, \tau], \tag{28}$$

or, equivalently (using Equations (8) and (21)),

$$k(\tau) = \lim_{f \to 0} S_0 \, \omega_0 \, e^{-\omega_0 \tau} \left[\cos(f \, \tau) + \frac{\omega_0}{f} \sin(f \, \tau) \right] \tag{29}$$

$$= S_0 \, \omega_0 \, e^{-\omega_0 \tau} \, [1 + \omega_0 \, \tau]. \tag{30}$$

This covariance function is also known as the Matérn-3/2 function (Rasmussen & Williams 2006). This model cannot be directly evaluated using celerite because, as we can see from Equation (21), the parameter b_j is infinite for $Q = 1/2$. The limit in Equation (29), however, suggests that an approximate Matérn-3/2 covariance can be computed with celerite by using a small value of f in Equation (29). The required value of f will depend on the specific data set and the precision requirements for the inference, so we encourage users to test this approximation in the context of their research before using it.

3. Finally, in the limit of large Q, the model approaches a high-quality oscillation with frequency ω_0 and covariance function

$$k(\tau) \approx S_0 \, \omega_0 \, Q \exp\left(-\frac{\omega_0 \, \tau}{2 \, Q} \right) \cos(\omega_0 \, \tau). \tag{31}$$

Figure 1 shows a plot of the PSD for these limits and several other values of Q. From this figure, it is clear that for $Q \leqslant 1/2$ the model has no oscillatory behavior and that for large Q the shape of the PSD near the peak frequency approaches a Lorentzian.

These special cases demonstrate that the stochastically driven SHO provides a physically motivated model that is flexible enough to describe a wide range of stellar variations, and we give some specific examples in Section 6. Low $Q \approx 1$ can capture granulation noise, and high $Q \gg 1$ is a good model for asteroseismic oscillations. In practice, we take a sum over oscillators with different values of Q, S_0, and ω_0 to give a sufficient accounting of the power spectrum of stellar time series. Since this kernel is exactly described by the exponential kernel, the likelihood (Equation (3)) can be evaluated for a time series with N measurements in $\mathcal{O}(N)$ operations using the *celerite* method described in the next section.

5. Semiseparable Matrices and Celerite

The previous two sections describe the *celerite* model and its physical interpretation. In this section, we derive a new scalable direct solver for covariance matrices generated by this model to compute the GP likelihood (Equation (3)) for large data sets. This method exploits the semiseparable structure of these matrices to compute their Cholesky factorization in $\mathcal{O}(N \, J^2)$ operations.

The relationship between the *celerite* model and semiseparable linear algebra was first recognized by Ambikasaran (2015) building on earlier work by Rybicki & Press (1995). Ambikasaran derived a scalable direct solver for any general semiseparable matrix and applied this method to *celerite* models with $b_j = 0$ and $d_j = 0$. The Ambikasaran (2015) method uses a factorization routine that can be applied to general semiseparable matrices. In other words, it can be applied to factorize matrices even if they are not positive definite. While preparing this paper, we generalized this method to include nonzero b_j and d_j, but all valid GP
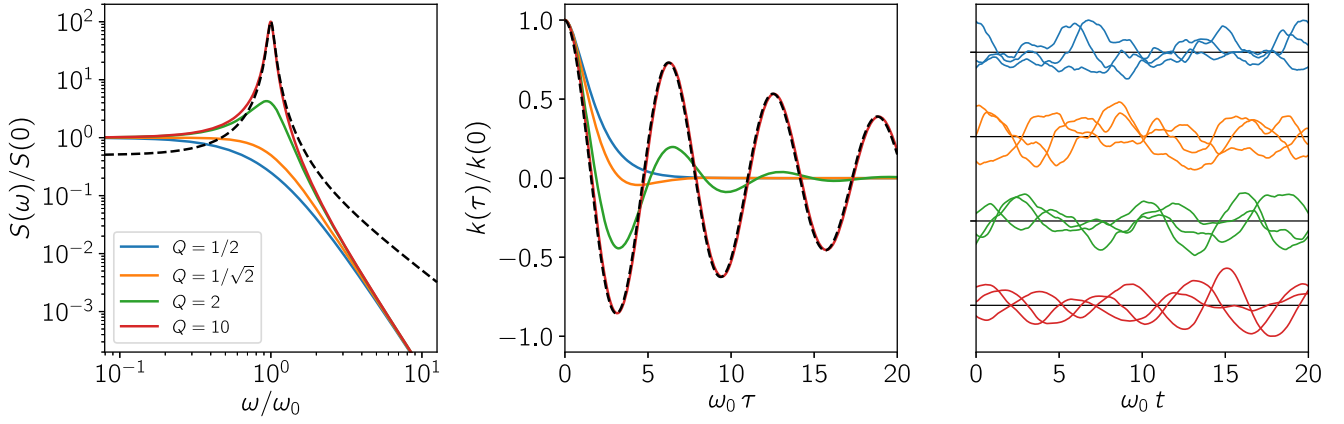
Figure 1. Left: power spectrum of a stochastically driven simple harmonic oscillator (Equation (20)) plotted for several values of the quality factor Q. For comparison, the dashed line shows the Lorentzian function from Equation (11) with $c_j = \omega_0/(2\,Q) = 1/20$ and normalized so that $S(d_j)/S(0) = 100$. Middle: corresponding autocorrelation functions with the same colors. Right: three realizations in arbitrary units from each model in the same colors.

covariance matrices are symmetric and positive definite, so we can exploit this constraint to derive a Cholesky factorization algorithm. It turns out that this algorithm is more than an order of magnitude faster than the previously published method when applied to *celerite* models. In this and the following sections we derive this novel factorization algorithm, and in Appendix A we discuss methods for ensuring that the covariance matrix for a *celerite* model is positive definite.

To start, we define a rank-R semiseparable matrix as a matrix K where the elements can be written as

$$
K_{n,m} = \begin{cases} \sum_{r=1}^{R} U_{n,r}\, V_{m,r} & \text{if } m < n \\ A_{n,n} & \text{if } m = n \\ \sum_{r=1}^{R} U_{m,r}\, V_{n,r} & \text{otherwise} \end{cases}
\tag{32}
$$

for some $N \times R$-dimensional matrices U and V and an $N \times N$-dimensional diagonal matrix A. This definition can be equivalently written as

$$
K = A + \mathrm{tril}(U\,V^T) + \mathrm{triu}(V\,U^T),
\tag{33}
$$

where the tril function extracts the strict lower triangular component of its argument and triu is the equivalent strict upper triangular operation. In this representation, rather than storing the full $N \times N$ matrix K, it is sufficient to only store the $N \times R$ matrices U and V and the N diagonal entries of A, for a total storage volume of $(2\,R + 1)N$ floating point numbers. Given a semiseparable matrix of this form, the matrix–vector products and matrix–matrix products can be evaluated in $\mathcal{O}(N)$ operations. We include a summary of the key operations in Appendix B, but the interested reader is directed to Vandebril et al. (2007) for more details about semiseparable matrices in general.

5.1. Cholesky Factorization of Positive-definite Semiseparable Matrices

For our purposes, we want to compute the log-determinant of K and solve linear systems of the form $K\,z = y$ for z, where K is a rank-R semiseparable positive-definite matrix. These can

both be computed using the Cholesky factorization of K:

$$
K = L\,D\,L^T,
\tag{34}
$$

where L is a lower triangular matrix with unit diagonal and D is a diagonal matrix. We use the *Ansatz* that L has the form

$$
L = I + \mathrm{tril}(U\,W^T),
\tag{35}
$$

where I is the identity, U is the matrix from above, and W is an unknown $N \times R$ matrix. Combining this with Equation (33), we find the equation

$$
\begin{aligned}
A &+ \mathrm{tril}(U\,V^T) + \mathrm{triu}(V\,U^T) \\
&= [I + \mathrm{tril}(U\,W^T)]\,D\,[I + \mathrm{tril}(U\,W^T)]^T \\
&= D + \mathrm{tril}(U\,W^T)D + D\,\mathrm{triu}(W\,U^T) \\
&+ \mathrm{tril}(U\,W^T)D\,\mathrm{triu}(W\,U^T),
\end{aligned}
\tag{36}
$$

which we can solve for W. Setting each element on the left-hand side of Equation (36) equal to the corresponding element on the right-hand side, we derive the following recursion relations:

$$
\begin{aligned}
S_{n,j,k} &= S_{n-1,j,k} + D_{n-1,n-1}\,W_{n-1,j}\,W_{n-1,k} \\
D_{n,n} &= A_{n,n} - \sum_{j=1}^{R} \sum_{k=1}^{R} U_{n,j}\,S_{n,j,k}\,U_{n,k} \\
W_{n,j} &= \frac{1}{D_{n,n}}\left[V_{n,j} - \sum_{k=1}^{R} U_{n,k}\,S_{n,j,k} \right],
\end{aligned}
\tag{37}
$$

where S_n is a symmetric $R \times R$ matrix and every element $S_{1,j,k} = 0$. The computational cost of one iteration of the update in Equation (37) is $\mathcal{O}(R^2)$, and this must be run N times—once for each row—so the total cost of this factorization scales as $\mathcal{O}(N\,R^2)$.

After computing the Cholesky factorization of this matrix K, we can also apply its inverse and compute its log-determinant in $\mathcal{O}(N\,R)$ and $\mathcal{O}(N)$ operations, respectively. The log-determinant of K is given by

$$
\ln \det K = \sum_{n=1}^{N} \ln D_{n,n},
\tag{38}
$$

where, since K is positive definite, $D_{n,n} > 0$ for all n.

The inverse of K can be applied using the relationship

$$z = K^{-1} y = (L^T)^{-1} D^{-1} L^{-1} y. \qquad (39)$$

First, we solve for $z' = L^{-1} y$ using forward substitution

$$f_{n,j} = f_{n-1,j} + W_{n-1,j} z'_{n-1}$$
$$z'_n = y_n - \sum_{j=1}^{R} U_{n,j} f_{n,j}, \qquad (40)$$

where $f_{0,j} = 0$ for all j. Then, using this result and backward substitution, we compute z:

$$g_{n,j} = g_{n+1,j} + U_{n+1,j} z_{n+1}$$
$$z_n = \frac{z'_n}{D_{n,n}} - \sum_{j=1}^{R} W_{n,j} g_{n,j}, \qquad (41)$$

where $g_{N+1,j} = 0$ for all j. The total cost of the forward and backward substitution passes scales as $\mathcal{O}(N\,R)$.

The algorithm derived here is a general method for factorizing, solving, and computing the determinant of positive-definite rank-R semiseparable matrices. As we will show below, this method can be used to compute Equation (3) for *celerite* models. In practice, when K is positive definite and this method is applicable, we find that this method is about a factor of 20 faster than an optimized implementation of the earlier, more general method from Ambikasaran (2015).

5.2. Scalable Computation of Celerite Models

The *celerite* model described in Section 3 can be exactly represented as a rank $R = 2J$ semiseparable matrix[12] where the components of the relevant matrices are

$$U_{n,2j-1} = a_j e^{-c_j t_n} \cos(d_j t_n) + b_j e^{-c_j t_n} \sin(d_j t_n)$$
$$U_{n,2j} = a_j e^{-c_j t_n} \sin(d_j t_n) - b_j e^{-c_j t_n} \cos(d_j t_n)$$
$$V_{m,2j-1} = e^{c_j t_m} \cos(d_j t_m)$$
$$V_{m,2j} = e^{c_j t_m} \sin(d_j t_m)$$
$$A_{n,n} = \sigma_n^2 + \sum_{j=1}^{J} a_j. \qquad (42)$$

A naïve implementation of the algorithm in Equation (37) for this system will result in numerical overflow and underflow issues in many realistic cases because both U and V have factors of the form $e^{\pm c_j t}$, where t can be any arbitrarily large real number. This issue can be avoided by reparameterizing the semiseparable representation from Equation (42). We define the following preconditioned variables:

$$\tilde{U}_{n,2j-1} = a_j \cos(d_j t_n) + b_j \sin(d_j t_n)$$
$$\tilde{U}_{n,2j} = a_j \sin(d_j t_n) - b_j \cos(d_j t_n)$$
$$\tilde{V}_{m,2j-1} = \cos(d_j t_m)$$
$$\tilde{V}_{m,2j} = \sin(d_j t_m)$$
$$\tilde{W}_{n,2j-1} = e^{-c_j t_n} W_{n,2j-1}$$
$$\tilde{W}_{n,2j} = e^{-c_j t_n} W_{n,2j}, \qquad (43)$$

and the new set of variables

$$\phi_{n,2j-1} = \phi_{n,2j} = e^{-c_j(t_n - t_{n-1})}, \qquad (44)$$

with the constraint

$$\phi_{1,2j-1} = \phi_{1,2j} = 0. \qquad (45)$$

Using this parameterization, we find the following numerically stable algorithm to compute the Cholesky factorization of a *celerite* model:

$$S_{n,j,k} = \phi_{n,j}\,\phi_{n,k}[S_{n-1,j,k} + D_{n-1,n-1} \tilde{W}_{n-1,j} \tilde{W}_{n-1,k}]$$
$$D_{n,n} = A_{n,n} - \sum_{j=1}^{2J} \sum_{k=1}^{2J} \tilde{U}_{n,j} S_{n,j,k} \tilde{U}_{n,k}$$
$$\tilde{W}_{n,j} = \frac{1}{D_{n,n}}\left[\tilde{V}_{n,j} - \sum_{k=1}^{2J} \tilde{U}_{n,k} S_{n,j,k}\right]. \qquad (46)$$

As before, a single iteration of this algorithm requires $\mathcal{O}(J^2)$ operations, and it is repeated N times for a total scaling of $\mathcal{O}(N\,J^2)$. Furthermore, $\tilde{V}_{n,j}$ and $A_{n,n}$ can be updated in place and used to represent $\tilde{W}_{n,j}$ and $D_{n,n}$, giving a total memory footprint of $(6J + 1)N + J(J - 1)/2$ floating point numbers.

The log-determinant of K can still be computed as in the previous section, but we must update the forward and backward substitution algorithms to account for the reparameterization in Equation (43). The forward substitution with the reparameterized variables is

$$f_{n,j} = \phi_{n,j}[f_{n-1,j} + \tilde{W}_{n-1,j} z'_{n-1}]$$
$$z'_n = y_n - \sum_{j=1}^{2J} \tilde{U}_{n,j} f_{n,j}, \qquad (47)$$

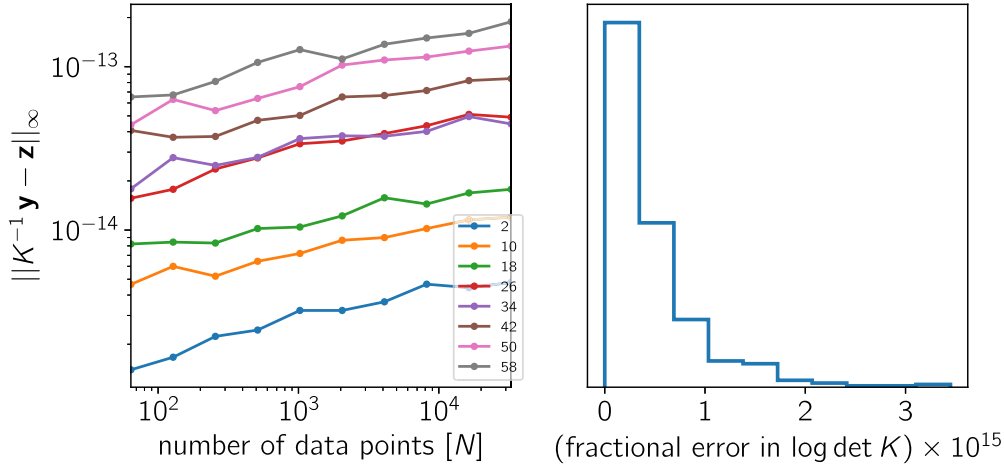where $f_{0,j} = 0$ for all j, and the backward substitution algorithm becomes

$$g_{n,j} = \phi_{n+1,j}[g_{n+1,j} + \tilde{U}_{n+1,j} z_{n+1}]$$
$$z_n = \frac{z'_n}{D_{n,n}} - \sum_{j=1}^{2J} \tilde{W}_{n,j} g_{n,j}, \qquad (48)$$

where $g_{N+1,j} = 0$ for all j. The only difference, when compared to Equations (40) and (41), is the multiplication by $\phi_{n,j}$ in the recursion relations for $f_{n,j}$ and $g_{n,j}$. As before, the computational cost of this solve scales as $\mathcal{O}(N\,J)$.

5.3. Performance

We have derived a novel and scalable method for computing the Cholesky factorization of the covariance matrices produced by *celerite* models. In this section, we demonstrate the numerical and computational performance of this algorithm.

First, we empirically confirm that the method solves systems of the form $K\,z = y$ to high numerical accuracy. The left panel of Figure 2 shows the L-infinity norm for the residual $z - K^{-1} y = z - K^{-1}Kz$ for a range of data set sizes N and numbers of terms J. For each pair of N and J we repeated the experiment 10 times with different randomly generated data sets and *celerite* parameters, and we averaged across experiments to remove some sampling noise from this figure. The numerical error introduced by our method increases weakly for larger data sets $\sim N^{0.15}$ and roughly linearly with J.

---

[12] We note that if some *celerite* terms are real ($b_j = 0$ and $d_j = 0$), then these terms only add one to the rank. Therefore, if a *celerite* model has J_{real} real and J_{complex} general terms, the semiseparable rank is only $R = J_{\text{real}} + 2J_{\text{complex}} = 2J - J_{\text{real}}$.

Figure 2. Left: maximum numerical error introduced by the algorithm described in the text as a function of the number of *celerite* terms J and data points N. Each line corresponds to a different J, and the error increases with J as shown in the legend. Each point in this figure is the average result across 10 systems with randomly sampled parameters. Right: distribution of fractional error on the log-determinant of K computed using the algorithm described in the text compared to the algorithm implemented in NumPy (Van Der Walt et al. 2011) for the same systems shown in the left panel. This histogram only includes comparisons for $N \leqslant 2048$ because standard methods become computationally expensive for larger systems.

For the experiments with small $N \leqslant 2048$, we also compared the log-determinant calculation to the result calculated using the general log-determinant function provided by the NumPy project (Van Der Walt et al. 2011), itself built on the LAPACK (Anderson et al. 1999) implementation in the Intel Math Kernel Library.[13] The right panel of Figure 2 shows a histogram of the fractional error in the log-determinant calculation for each of the 480 experiments with $N \leqslant 2048$. These results agree to about 10^{-15}, and the error only weakly increases with N and J.

We continue by measuring the real-world computational efficiency of this method as a function of N and J. This experiment was run on a 2013 MacBook Pro with a dual 2.6 GHz Intel Core i5 processor, but we find similar performance on other platforms. The celerite implementation used for this test is the Python interface with a C++ back end, but there is little overhead introduced by Python, so we find similar performance using the implementation in C++ and an independent implementation in Julia (Bezanzon et al. 2017). Figure 3 shows the computational cost of evaluating Equation (3) using celerite as a function of N and J. The empirical scaling as a function of data size matches the theoretical scaling of $\mathcal{O}(N)$ for all values of N, and the scaling with the number of terms is the theoretical $\mathcal{O}(J^2)$ for $J \gtrsim 10$, with some overhead for smaller models. For comparison, the computational cost of the same computation using the Intel Math Kernel Library, a highly optimized general dense linear algebra package, is shown as a dashed line, which is independent of J. Celerite is faster than the reference implementation for all but the smallest data sets with many terms, $J \gtrsim 64$ for $N \approx 512$, and J increasing with larger N.

5.4. How to Choose a Celerite Model

In any application of celerite—or any other GP model, for that matter—we must choose a kernel. A detailed discussion of model selection is beyond the scope of this paper, and the interested reader is directed to chapter 5 of Rasmussen & Williams (2006) for an introduction in the context of GPs, but

we include a summary and some suggestions specific to celerite here.

For many of the astrophysics problems that we will tackle using celerite, it is possible to motivate the kernel choice physically. We give three examples of this in the next section, and in these cases we have a physical description of the system that generated our data, we represent that system as a *celerite* model, and we use celerite for parameter estimation. For other projects, we might be interested in comparing different theories, and in those cases we would need to perform formal model selection.

Sometimes, it is not possible to choose a physically motivated kernel when we use a GP as an effective model. In these cases, we recommend starting with a mixture of stochastically driven damped SHOs (as discussed in Section 4) and selecting the number of oscillators using Bayesian model comparison, cross-validation, or another model selection technique. Despite the fact that it is less flexible than a general *celerite* kernel, we recommend the SHO model in most cases because it is once mean square differentiable

$$\left. \frac{dk_{\mathrm{SHO}}(\tau)}{d\tau} \right|_{\tau=0} = 0, \qquad (49)$$

giving smoother functions than the general *celerite* model, which is only mean square continuous.

For a similar class of models (we return to these below in Section 7) Kelly et al. (2014) recommend using an "information criterion," such as the Akaike information criterion (AIC) or the Bayesian information criterion (BIC), as a model selection metric. These criteria are easy to implement, and we use the BIC in Section 6.2, but each information criterion comes with caveats that should be considered in any application (see, e.g., Ivezić et al. 2014).

6. Examples

To demonstrate the application of celerite, we use it to perform posterior inference in two examples with realistic simulated data and three examples of real-world research problems. These examples have been chosen to justify the use

---

[13] https://software.intel.com/en-us/intel-mkl
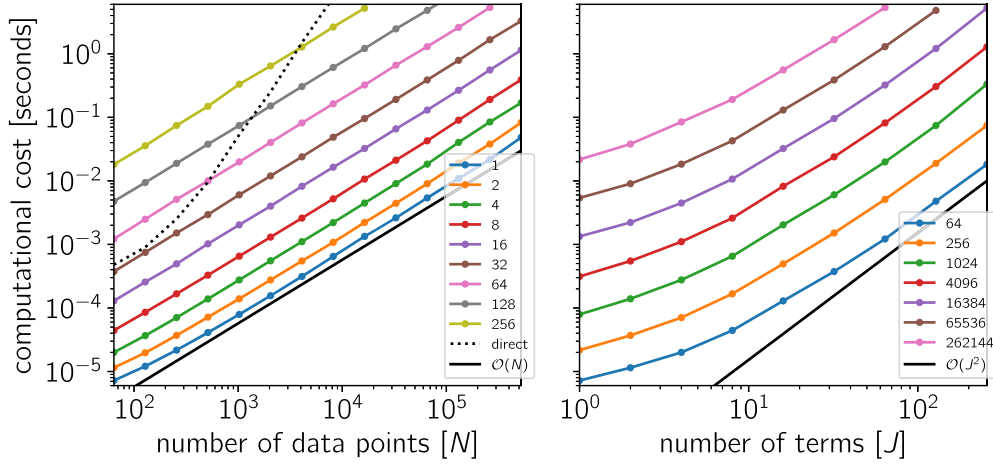
Figure 3. Benchmark showing the computational scaling of celerite using the Cholesky factorization routine for rank-2 J semiseparable matrices as described in the text. Left: cost of computing Equation (3) with a covariance matrix given by Equation (8) as a function of the number of data points N. The different lines show the cost for different numbers of terms J increasing from bottom to top. To guide the eye, the straight black line without points shows linear scaling in N. For comparison, the computational cost for the same model using a general Cholesky factorization routine implemented in the Intel MKL linear algebra package is shown as the dashed black line. Right: same information as in the left panel, plotted as a function of J for different values of N. Each line shows the scaling for a specific value of N increasing from bottom to top. The black line shows quadratic scaling in J.

of celerite for a range of research problems, but they are by no means exhaustive. These examples are framed in the time domain with a clear bias in favor of large homogeneous photometric surveys, but celerite can also be used for other one-dimensional problems such as spectroscopy, where wavelength—instead of time—is the independent coordinate (see Czekala et al. 2017, for a potential application).

In the first example, we demonstrate that celerite can be used to infer the power spectrum of a process when the data are generated from a *celerite* model. In the second example, we demonstrate that celerite can be used as an effective model even if the true process cannot be represented in the space of allowed models. This is an interesting example because, when analyzing real data, we rarely have any fundamental reason to believe that the data were generated by a GP model with a specific kernel. Even in these cases, GPs can be useful effective models, and celerite provides computational advantages over other GP methods.

The next three examples show celerite used to infer the properties of stars and exoplanets observed by NASA's Kepler mission. Each of these examples touches on an active area of research, so we limit our examples to be qualitative in nature and do not claim that celerite is the optimal method, but we hope that these examples encourage interested readers to investigate the applicability of celerite to their research.

In each example, the joint posterior probability density is given by

$$p(\boldsymbol{\theta}, \boldsymbol{\alpha} \mid \boldsymbol{y}, X) \propto p(\boldsymbol{y} \mid X, \boldsymbol{\theta}, \boldsymbol{\alpha}) p(\boldsymbol{\theta}, \boldsymbol{\alpha}), \quad (50)$$

where $p(\boldsymbol{y} \mid X, \boldsymbol{\theta}, \boldsymbol{\alpha})$ is the GP likelihood defined in Equation (3) and $p(\boldsymbol{\theta}, \boldsymbol{\alpha})$ is the joint prior probability density for the parameters. We sample each posterior density using MCMC and investigate the performance of the model when used to interpret the physical processes that generated the data. The specific parameters and priors are discussed in detail in each section, but we generally assume independent uniform priors with plausibly broad support.

While we do perform a convergence analysis in each case by computing the autocorrelation function and integrated auto-correlation time for the target chain (Sokal 1989; Goodman &

Weare 2010), the usual caveats associated with MCMC analysis apply when interpreting these results. For example, if there are multiple posterior modes that are not sampled by any chain, the autocorrelation analysis will not identify that the sampler has not converged. There are published sampling algorithms (Gregory 2005; Liu 2008; Feroz et al. 2009; Brewer et al. 2011; Kelly et al. 2014) that can, in some cases, be less sensitive to multimodality than the sampler that we use (Goodman & Weare 2010; Foreman-Mackey et al. 2013), but a comparison of sampling algorithms is beyond the scope of this paper, and celerite will also improve the computational efficiency of GP models when using the other inference algorithms.

6.1. Example 1: Recovery of a Celerite Process

In this first example, we simulate a data set using a known *celerite* process and fit it with celerite to show that valid inferences can be made in this idealized case. We simulate a small ($N = 200$) data set using a GP model with an SHO kernel (Equation (23)) with parameters $S_0 = 1$ ppm^2, $\omega_0 = e^2$ day^{-1}, and $Q = e^2$, where the units are arbitrarily chosen to simplify the discussion. We add further white noise with the amplitude $\sigma_n = 2.5$ ppm to each data point. The simulated data are plotted in the top left panel of Figure 4.

In this case, when simulating and fitting, we set the mean function $\mu_{\boldsymbol{\theta}}$ to zero—this means that the parameter vector $\boldsymbol{\theta}$ is empty—and the elements of the covariance matrix are given by Equation (23) with three parameters $\boldsymbol{\alpha} = (S_0, \omega_0, Q)$. We choose a proper separable prior for $\boldsymbol{\alpha}$ with log-uniform densities for each parameter as listed in Table 1.

To start, we estimate the maximum a posteriori (MAP) parameters using the L-BFGS-B nonlinear optimization routine (Byrd et al. 1995; Zhu et al. 1997) implemented by the SciPy project (Jones et al. 2001). The top left panel of Figure 4 shows the mean and standard deviation of the predicted noise-free data distribution from the MAP model overplotted as a blue contour on the simulated data, and the bottom panel shows the residuals away from this model. Since we are using the correct model to fit the data, it is reassuring
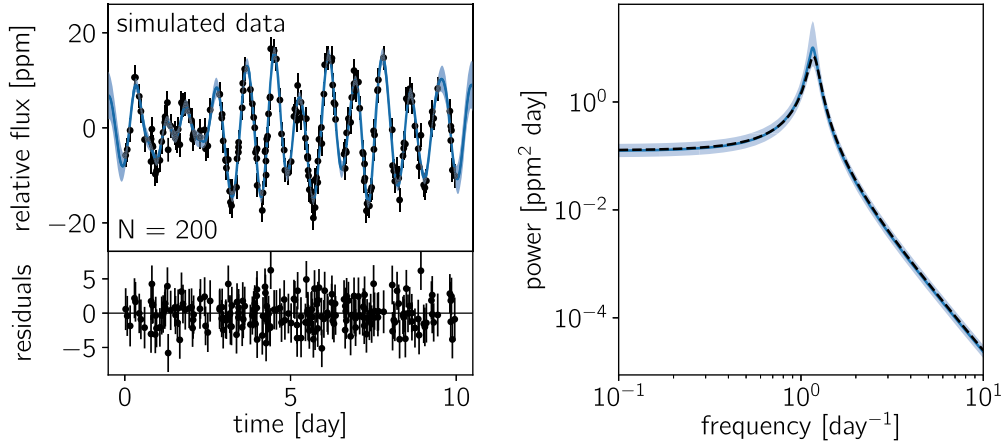
Figure 4. Top left: simulated data set (black error bars) and MAP model (blue contours). Bottom left: residuals between the mean predictive model and the data shown in the top left panel. Right: inferred PSD (the blue contours encompass 68% of the posterior mass) compared to the true PSD (dashed black line).

that the residuals appear qualitatively uncorrelated in this figure.

We then sample the joint posterior probability (Equation (50)) using emcee (Goodman & Weare 2010; Foreman-Mackey et al. 2013). We initialize 32 walkers[14] by sampling from a three-dimensional isotropic Gaussian centered on the MAP parameter vector and with a standard deviation of 10^{-4} in each dimension. We then run 500 steps of burn-in and 2000 steps of MCMC. To assess convergence, we estimate the integrated autocorrelation time of the chain averaged across parameters (Sokal 1989; Goodman & Weare 2010) and find that the chain results in 1737 effective samples.

Each sample in the chain corresponds to a model PSD, and we compare this posterior constraint on the PSD to the true spectral density in the right panel of Figure 4. In this figure, the true PSD is plotted as a dashed black line, and the numerical estimate of the posterior constraint on the PSD is shown as a blue contour indicating 68% of the MCMC samples. It is clear from this figure that, as expected, the inference correctly reproduces the true PSD.

6.2. Example 2: Inferences with the "Wrong" Model

In this example, we simulate a data set using a known GP model with a kernel outside of the support of a *celerite* process. This means that the true autocorrelation of the process can never be correctly represented by the model that we are using to fit, but we use this example to demonstrate that, at least in this case, valid inferences can still be made about the physical parameters of the model.

The data are simulated from a quasi-periodic GP with the kernel

$$k_{\text{true}}(\tau) = A \exp\left(-\frac{\tau^2}{2\,\lambda^2}\right) \cos\left(\frac{2\,\pi\,\tau}{P_{\text{true}}}\right), \quad (51)$$

where P_{true} is the fundamental period of the process. This autocorrelation structure corresponds to the power spectrum

---

[14] "Walker" is the name given by Goodman & Weare (2010) and Foreman-Mackey et al. (2013) to each parallel MCMC chain in the ensemble sampling algorithm used in emcee.

Table 1
Parameters and Priors for Example 1

Parameter	Prior
$\ln(S_0)$	$\mathcal{U}(-10, 10)$
$\ln(Q)$	$\mathcal{U}(-10, 10)$
$\ln(\omega_0)$	$\mathcal{U}(-10, 10)$

(Wilson & Adams 2013)

$$S_{\text{true}}(\omega) = \frac{\lambda A}{2}\left[\exp\left(-\frac{\lambda^2}{2}\left(\omega - \frac{2\,\pi}{P_{\text{true}}}\right)^2\right) \right.$$
$$\left. + \exp\left(-\frac{\lambda^2}{2}\left(\omega + \frac{2\,\pi}{P_{\text{true}}}\right)^2\right)\right], \quad (52)$$

which, for large values of ω, falls off exponentially. When compared to Equation (9)—which, for large ω, goes as ω^{-4} at most—it is clear that a *celerite* model can never exactly reproduce the structure of this process. That being said, we demonstrate that robust inferences can be made about P_{true} even with this effective model.

We generate a realization of a GP model with the kernel given in Equation (51) with $N = 100$, $A = 1$ ppm^2, $\lambda = 5$ days, and $P_{\text{true}} = 1$ day. We also add white noise with amplitude $\sigma = 0.5$ ppm to each data point. The top left panel of Figure 5 shows this simulated data set.

We then fit these simulated data using the product of two SHO terms (Equation (23)) where one of the terms has $S_0 = 1$ and $Q = 1/\sqrt{2}$ fixed. The kernel for this model is

$$k(\tau) = k_{\text{SHO}}(\tau; S_0, Q, \omega_1)$$
$$\times k_{\text{SHO}}(\tau; S_0 = 1, Q = 1/\sqrt{2}, \omega_2), \quad (53)$$

where k_{SHO} is defined in Equation (23) and the period of the process is $P = 2\,\pi/\omega_1$. We note that, using Equation (14), the product of two *celerite* terms can also be expressed using celerite.

We choose this functional form by comparing the BIC for a set of different models. In each model, we take the sum or product of J SHO terms, find the maximum likelihood model
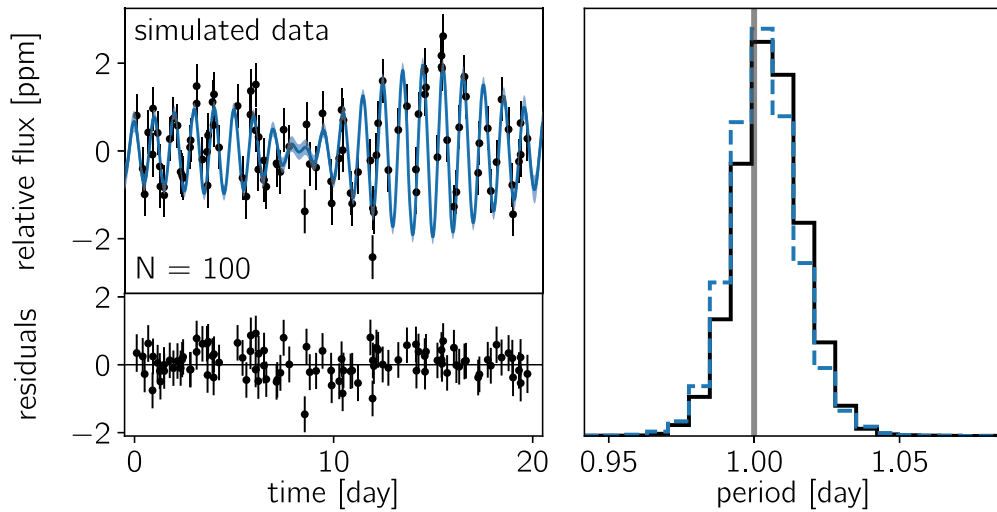
Figure 5. Top left: simulated data set (black error bars) and MAP model (blue contours). Bottom left: residuals between the mean predictive model and the data shown in the top left panel. Right: inferred period of the process. The true period is indicated by the vertical gray line, the posterior inference using the correct model is shown as the blue dashed histogram, and the inference made using the "wrong" effective model is shown as the black histogram.

using L-BFGS-B, and compute the BIC (Schwarz et al. 1978),

$$\mathrm{BIC} = -2\,\mathcal{L}^* + K \log N, \tag{54}$$

where $\mathcal{L}^*$ is the value of the likelihood at maximum, K is the number of parameters, and N is the number of data points. Figure 6 shows the value of the BIC for a set of products and sums of SHO terms, and the model that we choose has the lowest BIC.

As in the previous example, we set the mean function to zero and can, therefore, omit the parameters θ. Table 2 lists the proper log-uniform priors that we choose for each parameter in $\alpha = (S_0, Q, \omega_1, \omega_2)$. These priors, together with the GP likelihood (Equation (3)), fully specify the posterior probability density.

As above, we estimate the MAP parameters using L-BFGS-B and sample the posterior probability density using emcee. The top left panel of Figure 5 shows the conditional mean and standard deviation of the MAP model. The bottom left panel shows the residuals between the data and this MAP model, and even though this GP model is formally "wrong," there are no obvious correlations in these residuals.

To perform posterior inference, we initialize 32 walkers by sampling from the four-dimensional Gaussian centered on the MAP parameters with an isotropic standard deviation of 10^{-4}. We then run 500 steps of burn-in and 2000 steps of MCMC. To estimate the number of independent samples, we estimate the integrated autocorrelation time of the chain for the parameter ω_1—the parameter of primary interest—and find 1134 effective samples.

For comparison, we run the same number of steps of MCMC to sample the "correct" joint posterior density. For this reference inference, we use a GP likelihood with the kernel given by Equation (51) and choose log-uniform priors on each of the three parameters $\ln A/\mathrm{ppm}^2 \sim \mathcal{U}(-10, 10)$, $\ln \lambda/\mathrm{day} \sim \mathcal{U}(-10, 10)$, and $\ln P_{\mathrm{true}}/\mathrm{day} \sim \mathcal{U}(-10, 10)$.

The marginalized posterior inferences of the characteristic period of the process are shown in the right panel of Figure 5. The inference using the correct model is shown as a dashed blue histogram, and the inference made using the effective model is shown as a solid black histogram. These inferences
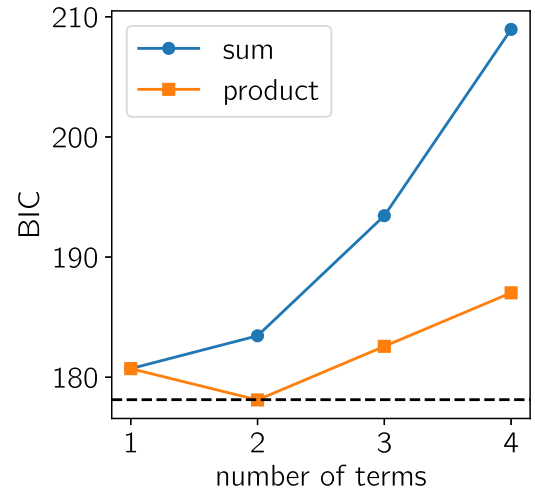
Figure 6. Bayesian information criterion (BIC) evaluated for the data from Figure 5 and different kernel choices. The x-axis shows the number of terms included in each model. The blue circles show models where the terms have been summed, and the orange squares indicate that the model is a product of terms. In the product models, the parameters S_0 and Q are held fixed at 1 and $1/\sqrt{2}$, respectively, for all but the first term. The dashed black line shows the minimum value of the BIC, and this corresponds to the model chosen in the text.

Table 2
Parameters and Priors for Example 2

Parameter	Prior
$\ln(S_0)$	$\mathcal{U}(-15, 5)$
$\ln(Q)$	$\mathcal{U}(-10, 10)$
$\ln(\omega_1)$	$\mathcal{U}(-10, 10)$
$\ln(\omega_2)$	$\mathcal{U}(-5, 5)$

are consistent with each other and with the true period used for the simulation (shown as a vertical gray line). This demonstrates that, in this case, celerite can be used as a computationally efficient effective model, and hints that this may be true in other problems as well.

6.3. Example 3: Stellar Rotation

A source of variability that can be measured from time series measurements of stars is rotation. The inhomogeneous surface of the star (spots, plage, etc.) imprints itself as quasi-periodic variations in photometric or spectroscopic observations (Dumusque et al. 2014). It has been demonstrated that for light curves with nearly uniform sampling the empirical autocorrelation function provides a reliable estimate of the rotation period of a star (McQuillan et al. 2013, 2014; Aigrain et al. 2015) and that a GP model with a quasi-periodic covariance function can be used to make probabilistic measurements even with sparsely sampled data (Angus et al. 2017). The covariance function used for this type of analysis has the form

$$k(\tau) = A \exp\left(-\frac{\tau^2}{2\,\ell^2} - \Gamma\,\sin^2\left(\frac{\pi\,\tau}{P_{\rm rot}}\right)\right), \qquad (55)$$

where $P_{\rm rot}$ is the rotation period of the star. GP modeling with the same kernel function has been proposed as a method of measuring the mean periodicity in quasi-periodic photometric time series in general (Wang et al. 2012). The key difference between Equation (55) and other quasi-periodic kernels is that it is positive for all values of τ. We construct a simple *celerite* covariance function with similar properties as follows:

$$k(\tau) = \frac{B}{2+C}\,e^{-\tau/L}\left[\cos\left(\frac{2\,\pi\,\tau}{P_{\rm rot}}\right) + (1+C)\right] \qquad (56)$$

for $B > 0$, $C > 0$, and $L > 0$. The covariance function in Equation (56) cannot exactly reproduce Equation (55), but since Equation (55) is only an effective model, Equation (56) can be used as a drop-in replacement for a significant gain in computational efficiency.

GPs have been used to measure stellar rotation periods for individual data sets (e.g., Littlefair et al. 2017), but the computational cost of traditional GP methods has hindered the industrial application to existing surveys like Kepler with hundreds of thousands of targets. The increase in computational efficiency and scalability provided by celerite opens the possibility of inferring rotation periods using GPs at the scale of existing and forthcoming surveys like Kepler, TESS, and LSST.

As a demonstration, we fit a *celerite* model with a kernel given by Equation (56) to a Kepler light curve for the star KIC 1430163. This star has a published rotation period of 3.88 ± 0.58 days, measured using traditional periodogram and autocorrelation function approaches applied to Kepler data from Quarters 0–16 (Mathur et al. 2014), covering about 4 yr.

We select about 180 days of contiguous observations of KIC 1430163 from Kepler. This data set has 6950 measurements, and, using a tuned linear algebra implementation,[15] a single evaluation of the likelihood requires over 8 s on a modern Intel CPU. This calculation, using celerite with the model in Equation (56), only takes ∼1.5 ms—a speed-up of more than three orders of magnitude per model evaluation.

We set the mean function μ_θ to zero, and the remaining parameters α and their priors are listed in Table 3. As with the earlier examples, we start by estimating the MAP parameters

---

[15] We use the Intel Math Kernel Library, https://software.intel.com/en-us/intel-mkl.

Table 3
Parameters and Priors for Example 3

Parameter	Prior
$\ln(B/{\rm ppt}^2)$	$\mathcal{U}(-10.0, 0.0)$
$\ln(L/{\rm day})$	$\mathcal{U}(1.5, 5.0)$
$\ln(P/{\rm day})$	$\mathcal{U}(-3.0, 5.0)$
$\ln(C)$	$\mathcal{U}(-5.0, 5.0)$

using L-BFGS-B and initialize 32 walkers by sampling from an isotropic Gaussian with a standard deviation of 10^{-5} centered on the MAP parameters. The left panels of Figure 7 show a subset of the data used in this example and the residuals away from the MAP predictive mean.

We run 500 steps of burn-in, followed by 5000 steps of MCMC using emcee. We estimate the integrated autocorrelation time of the chain for $\ln P_{\rm rot}$ and estimate that we have 2900 independent samples. These samples give a posterior constraint on the period of $P_{\rm rot} = 3.80 \pm 0.15$ days, and the marginalized posterior distribution for P is shown in the right panel of Figure 7. This result is in good agreement with the literature value with smaller uncertainties. A detailed comparison of GP rotation period measurements and the traditional methods is beyond the scope of this paper, but Angus et al. (2017) demonstrate that GP inferences are, at a population level, more reliable than other methods.

The total computational cost for this inference using celerite is about 4 CPU minutes. By contrast, the same inference using a general but optimized Cholesky factorization routine would require nearly 400 CPU hours. This speed-up enables probabilistic measurement of rotation periods using existing data from K2 and forthcoming surveys like TESS and LSST, where this inference will need to be run for at least hundreds of thousands of stars.

6.4. Example 4: Asteroseismic Oscillations

The asteroseismic oscillations of thousands of stars were measured using light curves from the Kepler mission (Gilliland et al. 2010; Chaplin et al. 2011, 2013; Huber et al. 2011; Stello et al. 2013), and asteroseismology is a key science driver for many of the upcoming large-scale photometric surveys (Rauer et al. 2014; Gould et al. 2015; Campante et al. 2016). Most asteroseismic analyses have been limited to high signal-to-noise oscillations because the standard methods use statistics of the empirical periodogram. These methods cannot formally propagate the measurement uncertainties to the constraints on physical parameters, and they instead rely on population-level bootstrap uncertainty estimates (Huber et al. 2009). More sophisticated methods that compute the likelihood function in the time domain scale poorly to large survey data sets (Brewer & Stello 2009; Corsaro & Ridder 2014).

Celerite alleviates these problems by providing a physically motivated probabilistic model that can be evaluated efficiently even for large data sets. In practice, we model the star as a mixture of stochastically driven SHOs where the amplitudes and frequencies of the oscillations are computed using a physical model, and we evaluate the probability of the observed time series using a GP where the PSD is a sum of terms given by Equation (20). This gives us a method for computing the likelihood function for the parameters of the physical model (e.g., $\nu_{\rm max}$ and $\Delta\nu$, or other more fundamental parameters)
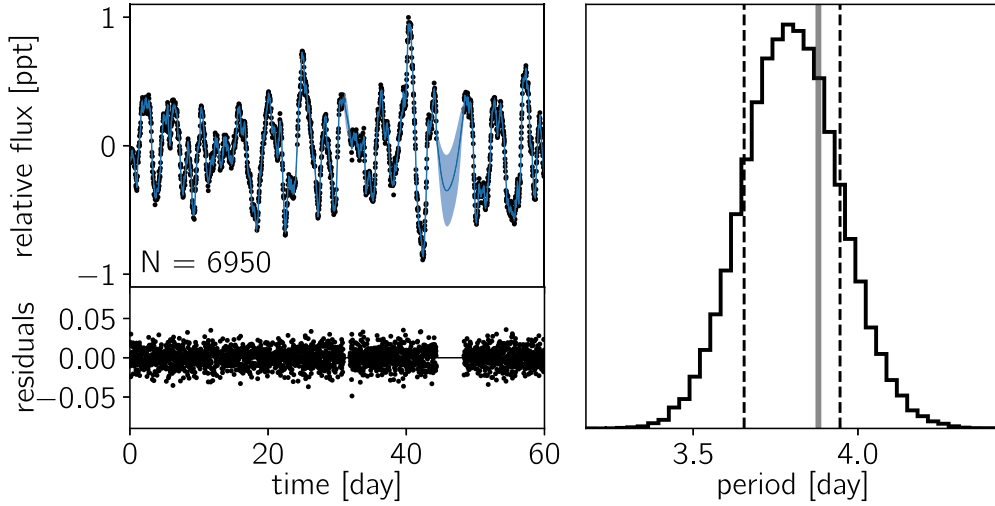
Figure 7. Inferred constraints on a quasi-periodic GP model using the covariance function in Equation (56) and two quarters of Kepler data. Top left: Kepler data (black points) and the MAP model prediction (blue curve) for a 60-day subset of the data used. The solid blue line shows the predictive mean, and the blue contours show the predictive standard deviation. Bottom left: residuals between the mean predictive model and the data shown in the top left figure. Right: posterior constraint on the rotation period of KIC 1430163 using the data set and model from Figure 7. The period is the parameter $P_{\rm rot}$ in Equation (56), and this figure shows the posterior distribution marginalized over all other nuisance parameters in Equation (56). The 1σ error bar on this measurement is indicated with vertical dashed lines. This result is consistent with the published rotation period made using the full Kepler baseline, shown as a vertical gray line (Mathur et al. 2014).

conditioned on the observed time series in $\mathcal{O}(N)$ operations. In other words, celerite provides a computationally efficient framework that can be combined with physically motivated models of stars and numerical inference methods to make rigorous probabilistic measurements of asteroseismic parameters in the time domain. This has the potential to push asteroseismic analysis to lower signal-to-noise data sets, and we hope to revisit this idea in a future paper.

To demonstrate the method, we use a simple heuristic model where the PSD is given by a mixture of eight components with amplitudes and frequencies specified by $\nu_{\rm max}$, $\Delta\nu$, and some nuisance parameters. The first term is used to capture the granulation "background" (Kallinger et al. 2014) using Equation (24) with two free parameters S_g and ω_g. The remaining seven terms are given by Equation (20), where Q is a nuisance parameter shared between terms, the frequencies are given by

$$\omega_{0,j} = 2\,\pi\,(\nu_{\rm max} + j\,\Delta\nu + \epsilon), \tag{57}$$

and the amplitudes are given by

$$S_{0,j} = \frac{A}{Q^2}\exp\left(-\frac{[j\,\Delta\nu + \epsilon]^2}{2\,W^2}\right), \tag{58}$$

where j is an integer running from -3 to 3 and ϵ, A, and W are shared nuisance parameters. Finally, we also fit for the amplitude of the white noise by adding a parameter σ in quadrature with the uncertainties given for the Kepler observations. All of these parameters and their chosen priors are listed in Table 4. As before, these priors are all log-uniform except for ϵ, where we use a zero-mean normal prior with a broad variance of 1 day^2 to break the degeneracy between $\nu_{\rm max}$ and ϵ. To build a more realistic model, this prescription could be extended to include more angular modes, or $\nu_{\rm max}$ and $\Delta\nu$ could be replaced by the fundamental physical parameters of the star.

To demonstrate the applicability of this model, we apply it to infer the asteroseismic parameters of the giant star KIC 11615890,

Table 4
Parameters and Priors for Example 4

| Parameter	Prior
$\ln(S_g/{\rm ppm}^2)$	$\mathcal{U}(-15, 15)$
$\ln(\omega_g/{\rm day}^{-1})$	$\mathcal{U}(-15, 15)$
$\ln(\nu_{\rm max}/\mu{\rm Hz})$	$\mathcal{U}(\ln(130), \ln(190))$
$\ln(\Delta\nu/\mu{\rm Hz})$	$\mathcal{U}(\ln(12.5), \ln(13.5))$
$\epsilon/{\rm day}^{-1}$	$\mathcal{N}(0, 1)$
$\ln(A/{\rm ppm}^2\,{\rm day})$	$\mathcal{U}(-15, 15)$
$\ln(Q)$	$\mathcal{U}(-15, 15)$
$\ln(W/{\rm day}^{-1})$	$\mathcal{U}(-3, 3)$
$\ln(\sigma/{\rm ppm})$	$\mathcal{U}(-15, 15)$

observed by the Kepler mission. The goal of this example is to show that, even for a low signal-to-noise data set with a short baseline, it is possible to infer asteroseismic parameters with formal uncertainties that are consistent with the parameters inferred with a much larger data set. Looking forward to TESS (Ricker et al. 2014; Campante et al. 2016), we measure $\nu_{\rm max}$ and $\Delta\nu$ using only 1 month of Kepler data and compare our results to the results inferred from the full 4 yr baseline of the Kepler mission. For KIC 11615890, the published asteroseismic parameters measured using several years of Kepler observations are (Pinsonneault et al. 2014)

$$\nu_{\rm max} = 171.94 \pm 3.62\ \mu{\rm Hz}$$
$$\text{and}\quad \Delta\nu = 13.28 \pm 0.29\ \mu{\rm Hz}. \tag{59}$$

Unlike typical giants, KIC 11615890 is a member of a class of stars where the dipole ($\ell = 1$) oscillation modes are suppressed by strong magnetic fields in the core (Stello et al. 2016). This makes this target simpler for the purposes of this demonstration because we can neglect the $\ell = 1$ modes and the model proposed above will be an effective model for the combined signal from the $\ell = 0$ and 2 modes.

For this demonstration, we randomly select a month-long segment of PDC Kepler data (Smith et al. 2012; Stumpe et al. 2012). Unlike the previous examples, the posterior distribution
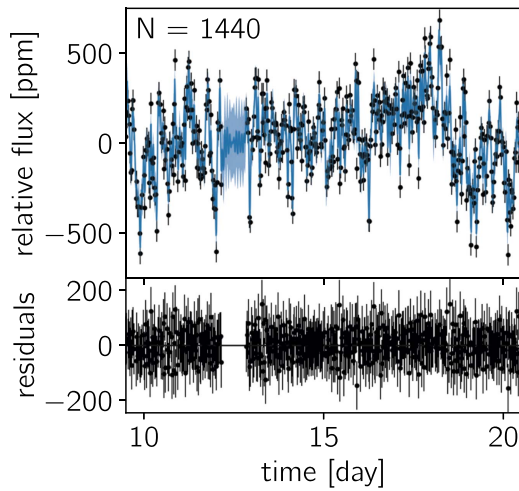
Figure 8. Top: Kepler data (black points) and the MAP model prediction (blue curve) for a 10-day subset of the month-long data set that was used for the fit. The solid blue line shows the predictive mean, and the blue contours show the predictive standard deviation. Bottom: residuals between the mean predictive model and the data shown in the top figure.

is sharply multimodal, and naïvely maximizing the posterior using L-BFGS-B is not practical. Instead, we start by estimating the initial values for ν_{max} and $\Delta\nu$ using only the month-long subset of data and following the standard procedure described by Huber et al. (2009). We then run a set of L-BFGS-B optimizations with values of ν_{max} selected in a logarithmic grid centered on our initial estimate of ν_{max} and initial values of $\Delta\nu$ computed using the empirical relationship between these two quantities (Stello et al. 2009). This initialization procedure is sufficient for this example, but the general application of this method will require a more sophisticated prescription.

Figure 8 shows a 10-day subset of the data set used for this example. The MAP model is overplotted on these data, and the residuals away from the mean prediction of this model are shown in the bottom panel of Figure 8. There is no obvious structure in these residuals, lending some credibility to the model specification.

We initialize 32 walkers by sampling from an isotropic Gaussian centered on the MAP parameters (the full set of parameters and their priors are listed in Table 4), run 5000 steps of burn-in, and run 15,000 steps of MCMC using emcee. We estimate the mean autocorrelation time for the chains of $\ln \nu_{max}$ and $\ln \Delta\nu$ and find 1443 effective samples from the marginalized posterior density. Figure 9 shows the marginalized density for ν_{max} and $\Delta\nu$ compared to the results from the literature. This result is consistent within the published error bars, and the posterior constraints are tighter than the published results. The top two panels of Figure 10 show the Lomb–Scargle periodogram (VanderPlas 2017) estimator for the power spectrum based on the full 4 yr of Kepler data and the month-long subset used in our analysis. The bottom panel of Figure 10 shows the posterior inference of the PSD using the celerite and the model described here. Despite only using 1 month of data, the celerite inference captures the salient features of the power spectrum, and it is qualitatively consistent with the standard estimator applied to the full baseline. All asteroseismic analyses are known to have substantial systematic and method-dependent uncertainties (Verner et al. 2011), so
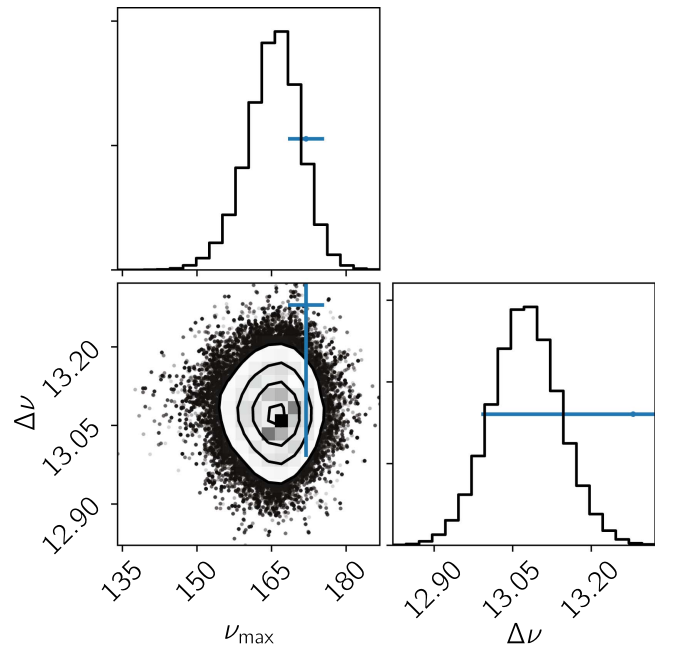
Figure 9. Probabilistic constraints on ν_{max} and $\Delta\nu$ from the inference shown in Figure 8 compared to the published value (blue error bars) based on several years of Kepler observations (Pinsonneault et al. 2014). The two-dimensional contours show the 0.5σ, 1σ, 1.5σ, and 2σ credible regions in the marginalized planes, and the histograms along the diagonal show the marginalized posterior for each parameter.

further experiments would be needed to fully assess the reliability of this specific model.

For this example, about 10 CPU minutes were required to run the MCMC. This is more computationally intensive than traditional methods of measuring asteroseismic oscillations, but it is much cheaper than the same analysis using a general direct GP solver. For comparison, we estimate that repeating this analysis using a general Cholesky factorization implemented as part of a tuned linear algebra library[16] would require about 15 CPU hours. An in-depth discussion of the benefits of rigorous probabilistic inference of asteroseismic parameters in the time domain is beyond the scope of this paper, but we hope to revisit this opportunity in the future.

6.5. Example 5: Exoplanet Transit Fitting

This example is different from all the previous examples—both simulated and real—because, in this case, we do not set the mean function μ_θ to zero. Instead, we make inferences about μ_θ because it is a physically significant function and the parameters are fundamental properties of the system. This is an example using real data—because we use the light curve from Section 6.3—but we multiply these data by a simulated transiting exoplanet model with known parameters. This allows us to show that we can recover the true parameters of the planet even when the transit signal is superimposed on the real variability of a star. GP modeling has been used extensively for this purpose throughout the exoplanet literature (e.g., Dawson et al. 2014; Barclay et al. 2015; Evans et al. 2015; Foreman-Mackey et al. 2016; Grunblatt et al. 2016).

In Equation (3) the physical parameters of the exoplanet are called θ, and in this example the mean function $\mu_\theta(t)$ is a

---

[16] We use the Intel Math Kernel Library, https://software.intel.com/en-us/intel-mkl.
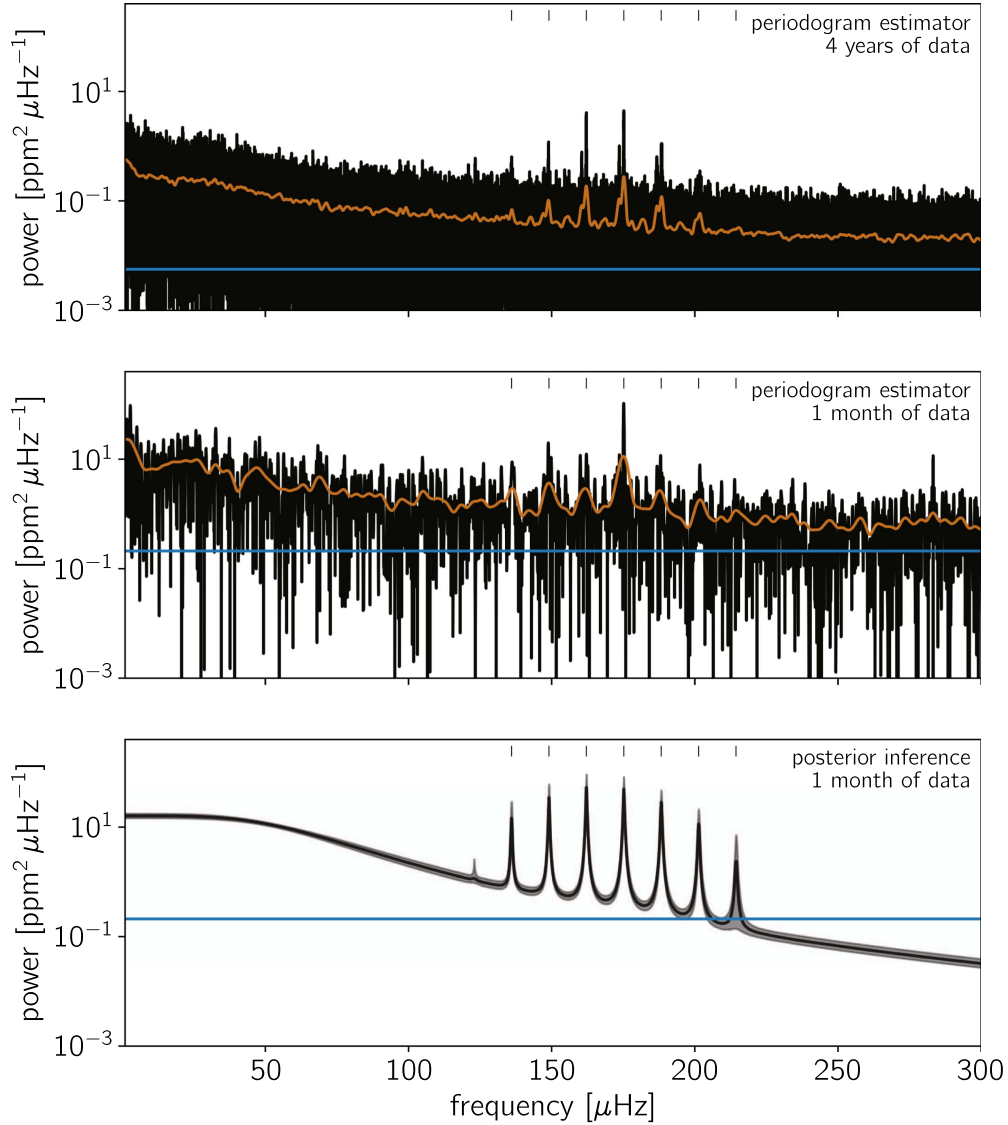
Figure 10. Comparison between the Lomb–Scargle estimator of the PSD and the posterior inference of the PSD as a mixture of stochastically driven simple harmonic oscillators. Top: periodogram of the Kepler light curve for KIC 11615890 computed on the full 4 yr baseline of the mission. The orange line shows a smoothed periodogram, and the blue line indicates the level of the measurement uncertainties. Middle: same periodogram computed using about a month of data. Bottom: power spectrum inferred using the mixture of SHOs model described in the text and only 1 month of Kepler data. The black line shows the median of posterior PSD, and the gray contours show the 68% credible region.

limb-darkened transit light curve (Mandel & Agol 2002; Foreman-Mackey & Morton 2016) and the parameters θ are the orbital period P_{orb}, the transit duration T, the phase or epoch t_0, the impact parameter b, the radius of the planet in units of the stellar radius $R_P/R_\star$, the baseline relative flux of the light curve f_0, and two parameters describing the limb-darkening profile of the star (Claret & Bloemen 2011; Kipping 2013). As in Section 6.3, we model the stellar variability using a GP model with a kernel given by Equation (56) and fit for the parameters of the exoplanet θ and the stellar variability α simultaneously. The full set of parameters α and θ is listed in Table 5, along with their priors and the true values for θ.

We take a 20-day segment of the Kepler light curve for KIC 1430163 ($N = 1000$) and multiply it by a simulated transit model with the parameters listed in Table 5. Using

these data, we maximize the joint posterior defined by the likelihood in Equation (3) and the priors in Table 5 using L-BFGS-B for all the parameters α and θ simultaneously. The top panel of Figure 11 shows the data including the simulated transit as black points, with the MAP model prediction overplotted in blue. The bottom panel of Figure 11 shows the "detrended" light curve, where the MAP model has been subtracted and the MAP mean model μ_θ has been added back in. For comparison, the transit model is overplotted in the bottom panel of Figure 11, and we see no obvious correlations in the residuals.

Sampling 32 walkers from an isotropic Gaussian centered on the MAP parameters, we run 10,000 steps of burn-in and 30,000 steps of production MCMC using emcee. We estimate the integrated autocorrelation time for the $\ln(P_{\mathrm{orb}})$ chain and

Table 5
Parameters, Priors, and (if Known) the True Values Used for the Simulation in Example 5

Parameter	Prior	True Value
	Kernel: α	
$\ln(B/\text{ppt}^2)$	$\mathcal{U}(-10.0, 0.0)$	...
$\ln(L/\text{day})$	$\mathcal{U}(1.5, 5.0)$	...
$\ln(P_{\text{rot}}/\text{day})$	$\mathcal{U}(-3.0, 5.0)$	...
$\ln(C)$	$\mathcal{U}(-5.0, 5.0)$	...
$\ln(\sigma/\text{ppt})$	$\mathcal{U}(-5.0, 0.0)$	...
	Mean: θ	
f_0/ppt	$\mathcal{U}(-0.5, 0.5)$	0
$\ln(P_{\text{orb}}/\text{day})$	$\mathcal{U}(\ln(7.9), \ln(8.1))$	$\ln(8)$
$\ln(R_p/R_\star)$	$\mathcal{U}(\ln(0.005), \ln(0.1))$	$\ln(0.015)$
$\ln(T/\text{day})$	$\mathcal{U}(\ln(0.4), \ln(0.6))$	$\ln(0.5)$
t_0/day	$\mathcal{U}(-0.1, 0.1)$	0
b	$\mathcal{U}(0, 1.0)$	0.5
q_1	$\mathcal{U}(0, 1)$	0.5
q_2	$\mathcal{U}(0, 1)$	0.5

find 1490 effective samples across the full chain. Figure 12 shows the marginalized posterior constraints on the physical properties of the planet compared to the true values. Even though the *celerite* representation of the stellar variability is only an effective model, the inferred distributions for the physical properties of the planet θ are consistent with the true values. This promising result suggests that celerite can be used as an effective model for transit inference for a significant gain in computational tractability. Even for this example with small N, the cost of computing the likelihood using celerite is nearly two orders of magnitude faster than the same computation using a general direct solver.

6.6. Summary

In the previous subsections, we demonstrate five potential use cases for celerite. These examples span a range of data sizes and model complexities, and the last three are based on active areas of research with immediate real-world applicability. The sizes of these data sets are modest on the scale of some existing and forthcoming survey data sets because we designed the experiments to be easily reproducible, but, even in these cases, the inferences made here would be intractable without substantial computational investment.

Table 6 lists the specifications of each example. In this table, we list the computational cost required for a single likelihood evaluation using a general Cholesky factorization and the cost of computing the same likelihood using celerite. Combining this with the total number of model evaluations required to run the MCMC algorithm to convergence (this number is also listed in Table 6), we can estimate the total CPU time required in each case. While the exact computational requirements for any problem also depend on the choice of inference algorithm and the specific precision requirements, this table provides a rough estimate of what to expect for projects at these scales.

7. Comparisons to Other Methods

There are other methods of scaling GP models to large data sets, and in this section we draw comparisons between celerite and other popular methods. Scalable GP methods tend to fall into two categories: approximate and restrictive. Celerite falls into the latter category because, while the method is exact, it requires a specific choice of stationary kernel function and can only be used in one dimension.

The continuous autoregressive moving average (CARMA) models introduced to the astrophysics literature by Kelly et al. (2014) share many features with celerite. CARMA models are restricted to one-dimensional problems, and the likelihood function for a CARMA model can be solved in $\mathcal{O}(N\,J^2)$ using a Kalman filter (Kelly et al. 2014). The kernel function for a CARMA (J, K) model is

$$k_{\text{CARMA}}(\tau) = \sum_{j=1}^{2J} A_j \exp(r_j\,\tau), \qquad (60)$$

where

$$A_j = \sigma^2 \frac{\left[\sum_{k=0}^{K}\beta_k\,(r_j)^k\right]\left[\sum_{k=0}^{K}\beta_k\,(-r_j)^k\right]}{-2\,\text{Re}(r_j)\displaystyle\prod_{k=1, k\neq j}^{J}(r_k - r_j)(r_k{}^* + r_j)} \qquad (61)$$

and σ, $\{r_j\}_{j=1}^{J}$ and $\{\beta_k\}_{k=1}^{K}$ are parameters of the model; σ is real, while the others are complex. Comparing Equation (60) to Equation (7), we can see that every CARMA model corresponds to an equivalent *celerite* model and the parameters a_j, b_j, c_j, and d_j can be easily computed analytically.[17] The inverse statement is not as simple. In practice, this means that celerite could be trivially used to compute any CARMA model. Using the CARMA solver to compute a *celerite* model, however, requires solving Equation (61) numerically for a given set of $\{A_j\}_{j=1}^{J}$.

The computational scaling of CARMA models is also $\mathcal{O}(N\,J^2)$ using the Kalman filter method (Kelly et al. 2014), but we find that, in practice, this method is more computationally expensive than celerite but has a smaller memory footprint. Figure 13 shows the scaling of the Kalman filter solver for the systems shown in Figure 3. Comparing these figures, we find that the Kalman filter solver is slower than celerite for all the systems we tested with an average difference of about an order of magnitude.

Another benefit of celerite compared to Kalman filter solvers is the fact that it is an algebraic solver for the relevant matrix equations instead of an algorithm to directly compute Equation (3). This means that the inverse covariance matrix K^{-1} can be applied to general vectors and matrices in $\mathcal{O}(N)$, while reusing the factorization. This is a crucial feature when GP models are combined with linear regression (e.g., Luger et al. 2017).

Another popular GP method uses the fact that, in the limit of evenly spaced data and homoscedastic uncertainties, the covariance matrix is "Toeplitz" (e.g., Dillon et al. 2013). There are exact methods for solving Toeplitz matrix equations that scale as $\mathcal{O}(N \log N)$ and methods for computing

---

[17] Each CARMA term with complex parameters A_j and r_j corresponds to a *celerite* term with parameters $a_j = 2\,\text{Re}(A_j)$, $b_j = 2\,\text{Im}(A_j)$, $c_j = -\text{Re}(r_j)$, and $d_j = -\text{Im}(r_j)$.
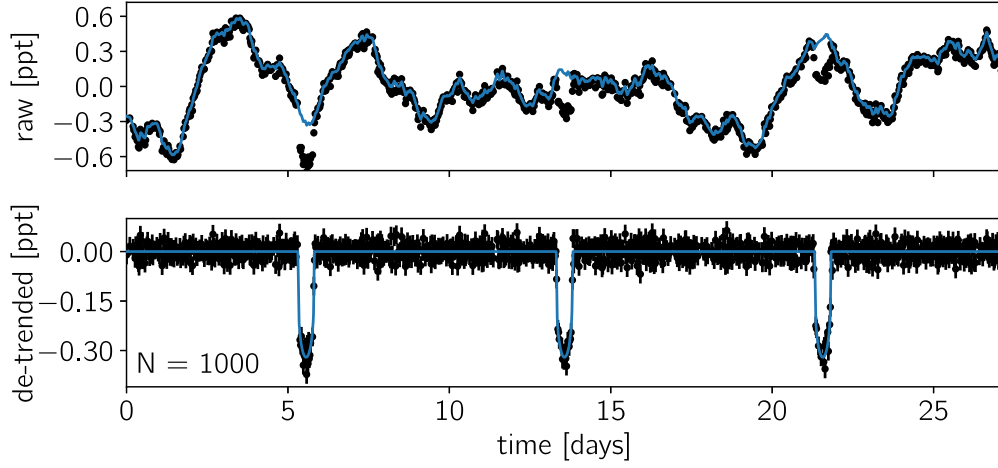
Figure 11. Top: month-long segment of the Kepler light curve for KIC 1430163 with a synthetic transit model injected (black points) and the MAP model for the stellar variability (blue line). Bottom: MAP "detrending" of the data in the top panel. In this panel, the MAP model for the stellar variability has been subtracted to leave only the transits. The detrended light curve is shown by black error bars, and the MAP transit model is shown as a blue line.
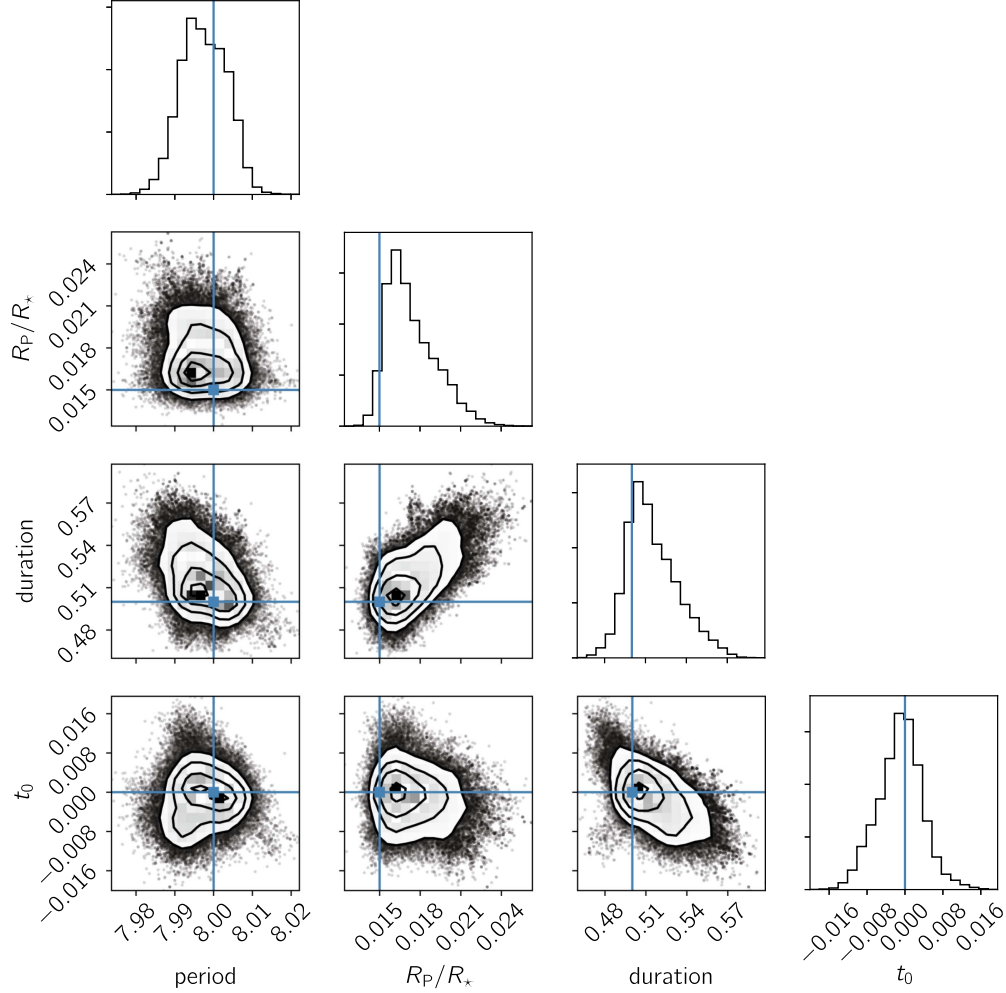
Figure 12. Marginalized posterior constraints on the physical parameters of the planet transit in the light curve shown in the top panel of Figure 11. The two-dimensional contours show the 0.5σ, 1σ, 1.5σ, and 2σ credible regions in the marginalized planes, and the histograms along the diagonal show the marginalized posterior for each parameter. The true values used in the simulation are indicated by blue lines. For each parameter, the inference is consistent with the true value.

determinants exactly in $\mathcal{O}(N^2)$ or approximately in $\mathcal{O}(N \log N)$ (Wilson 2014). The Toeplitz method is, in some ways, more flexible than celerite because it can be used with any stationary kernel, but it requires uniformly spaced data and the scaling is

worse than celerite, so it is less efficient when applied to large data sets.

Carter & Winn (2009) improved the scaling of Toeplitz methods by introducing a wavelet-based method for computing

Table 6
Computational Cost and Convergence Stats for Each Example

	N	J	Direct[a] (ms)	Celerite[b] (ms)	Dimension[c]	Evaluations[d]	$N_{\rm eff}$[e]
1	200	1	2.85	0.26	3	80,000	1737
2	100	2	0.67	0.36	4	80,000	1134
3	6950	2	8119.11	1.47	4	176,000	2900
4	1440	8	828.93	2.74	9	640,000	1443
5	1000	2	88.30	0.96	13	1,280,000	1490

Notes.
[a] The computational cost of computing the GP model using the general Cholesky factorization routine implemented in the Intel MKL.
[b] The computational cost of computing the GP model using celerite.
[c] The total number of parameters in the model.
[d] The total number of evaluations of the model to run MCMC to convergence.
[e] The effective number of independent samples estimated by computing the integrated autocorrelation time of the chain.
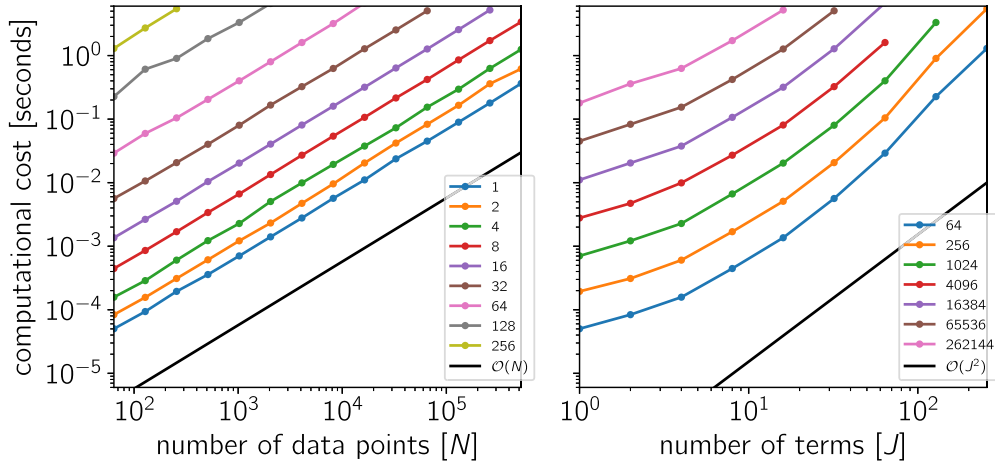
Figure 13. Same as Figure 3, but using an optimized C++ implementation of the Kalman filter method developed by Kelly et al. (2014). Comparing this figure to Figure 3, we see that celerite is about an order of magnitude faster in all cases. Left: cost of computing Equation (3) with a covariance matrix given by Equation (8) as a function of the number of data points N. The different lines show the cost for different numbers of terms J increasing from bottom to top. To guide the eye, the straight black line without points shows linear scaling in N. Right: same information plotted as a function of J for different values of N. Each line shows the scaling for a specific value of N increasing from bottom to top. The black line shows quadratic scaling in J.

a GP likelihood with $\mathcal{O}(N)$ scaling. This method has been widely applied in the context of exoplanet transit characterization, but it requires evenly spaced observations, and the power spectrum of the process must have the form $S(\omega) \propto \omega^{-1}$ to gain the computational advantage. This wavelet method has been demonstrated to improve parameter estimation for transiting exoplanets (Carter & Winn 2009), but these strict requirements make this method applicable for only a limited set of use cases.

Another GP method that has been used extensively in astronomy is the hierarchical off-diagonal low-rank (HODLR) solver (Ambikasaran et al. 2016). This method exploits the fact that many commonly used kernel functions produce "smooth" matrices to approximately compute the GP likelihood with the scaling $\mathcal{O}(N \log^2 N)$. This method has the advantage that, unlike celerite, it can be used with any kernel function, but, in practice, the cost can still prove to be prohibitively high for multidimensional inputs. The proportionality constant in the $N \log^2 N$ scaling of the HODLR method is a function of the specific kernel, and we find—using the george software package (Foreman-Mackey et al. 2014; Ambikasaran et al. 2016)—that this scales approximately linearly with J, but it requires substantial overhead, making it much slower than celerite for all the models we tested. For large $J \gtrsim 256$

and small $N \lesssim 1000$, we find that george can approximately evaluate *celerite* models with the same or less computational cost than celerite, but that celerite is faster—and exact—in all other parts of parameter space.

The structured kernel interpolation (SKI/KISS-GP; Wilson & Nicisch 2015) framework is another approximate method that can be used to scale GPs to large data sets. This method uses fast interpolation of the kernel into a space where Toeplitz or Kronecker structure can be used to scale inference and prediction. The SKI/KISS-GP framework can be applied to scale GP inference with a wide range of kernels, but the computational cost will depend on the specific data set, model, and precision requirements for the approximation. The SKI method has an impressive $\mathcal{O}(1)$ cost for test-time predictions, and it is interesting to consider how this could be applied to *celerite* models.

Many other approximate methods for scaling GP inference exist (see, e.g., Wilson et al. 2015, and references therein), and we make no attempt to make our discussion exhaustive. The key takeaway here is that celerite provides an *exact* method for GP inference for a specific class of one-dimensional kernel functions. Furthermore, since *celerite* models can be interpreted as a mixture of stochastically driven, damped SHOs, they are a

physically motivated choice of covariance function in many astronomical applications.

8. Discussion

GP models have been fruitfully applied to many problems in astronomical data analysis, but the fact that the computational cost generally scales as the cube of the number of data points has limited their use to small data sets with $N \lesssim 1000$. With the linear scaling of celerite we envision that the application of GPs will be expanded to the existing and forthcoming large astronomical time domain surveys such as Kepler (Borucki et al. 2010), K2 (Howell et al. 2014), TESS (Ricker et al. 2014), LSST (Ivezić et al. 2008), WFIRST (Spergel et al. 2015), and PLATO (Rauer et al. 2014). Despite the restrictive form of the celerite kernel, with a sufficient number of components it is flexible enough to describe a wide range of astrophysical variability. In fact, the relation of the celerite kernel to the damped, stochastically driven harmonic oscillator matches simple models of astrophysical variability and makes the parameterization interpretable in terms of resonant frequency, amplitude, and quality factor.

Our background is in studying transiting exoplanets, a field that has only recently begun to adopt GP methods for analyzing the noise in stellar light curves and radial velocity data sets when detecting or characterizing transiting planets (e.g., Carter & Winn 2009; Gibson et al. 2012; Haywood et al. 2014; Barclay et al. 2015; Evans et al. 2015; Rajpaul et al. 2015; Aigrain et al. 2016; Foreman-Mackey et al. 2016; Grunblatt et al. 2016; Luger et al. 2016). All of these analyses have been limited to small data sets or restrictive kernel choices, but celerite weakens these requirements by providing a scalable method for computing the likelihood and a physical motivation for the choice of kernel. As higher signal-to-noise observations of transiting exoplanet systems are obtained, the effects of stellar variability will more dramatically impact the correct inference of planetary transit parameters. We predict that celerite will be important for scaling methods of transit detection (Foreman-Mackey et al. 2016; Pope et al. 2016), transit timing (Agol et al. 2005; Holman 2005), transit spectroscopy (Brown 2001), Doppler beaming (Loeb & Gaudi 2003; Zucker et al. 2007), tidal distortion (Zucker et al. 2007), and phase functions (Knutson et al. 2007; Zucker et al. 2007) to the large astronomical time domain surveys of the future.

8.1. Other Applications and Limitations

Beyond these applications to model stellar variability, the method is generally applicable to other one-dimensional GP models. Accreting black holes show time series that may be modeled using a GP (Kelly et al. 2014); indeed, this was the motivation for the original technique developed by Rybicki & Press (1992, 1995). This approach may be broadly used for characterizing quasar variability (MacLeod et al. 2010), measuring time lags with reverberation mapping (Zu et al. 2011; Pancoast et al. 2014), modeling time delays in multiply-imaged gravitationally lensed systems (Press & Rybicki 1998), characterizing quasi-periodic variability in a high-energy source (McAllister et al. 2016), or classification of variable objects (Zinn et al. 2016). We expect that there are also applications beyond astronomy.

The celerite formalism can also be used for power spectrum estimation and quantification of its uncertainties. A mixture of celerite terms can be used to perform nonparametric probabilistic inference of the power spectrum despite unevenly spaced data with heteroscedastic noise (for examples of this procedure, see Wilson & Adams 2013; Kelly et al. 2014). This type of analysis will be limited by the quadratic scaling of celerite . with the number of terms J, but this limits existing methods as well (Kelly et al. 2014).

There are many data analysis problems where celerite will not be immediately applicable. In particular, the restriction to one-dimensional problems is significant. There are many examples of multidimensional GP modeling in the astrophysics literature (recent examples from the field of exoplanet characterization include Haywood et al. 2014; Rajpaul et al. 2015; Aigrain et al. 2016), where celerite cannot be used to speed up the analysis. It is plausible that an extension can be derived to tackle some multidimensional problems with the right structure—simultaneous parallel time series, for example—and we hope to revisit this possibility in future work.

8.2. Code Availability

Alongside this paper, we have released a well-tested and documented open-source software package that implements the method and all of the examples discussed in these pages. This software is available on GitHub https://github.com/dfm/celerite[18] and Zenodo, and it is made available under the MIT license.

---

[18] This version of the paper was generated with git commit `24e321d8` (2017 September 22).

Appendix A
Ensuring Positive Definiteness for Celerite Models

For a GP kernel to be valid, it must produce a positive-definite covariance matrix for all input coordinates. For stationary kernels, this is equivalent—by Bochner's theorem (see Section 4.2.1 in Rasmussen & Williams 2006)—to requiring that the kernel be the Fourier transform of a positive finite measure. This means that the power spectrum of a positive-definite kernel must be positive for all frequencies. This result is intuitive because, since the power spectrum of a process is defined as the expected squared amplitude of the Fourier transform of the time series, it must be non-negative.

Using Equation (9), we find that for a single *celerite* term, this requirement is met when

$$\frac{(a_j c_j + b_j d_j)(c_j^2 + d_j^2) + (a_j c_j - b_j d_j)\omega^2}{\omega^4 + 2(c_j^2 - d_j^2)\omega^2 + (c_j^2 + d_j^2)^2} > 0. \quad (62)$$

The denominator is positive for all $c_j \neq 0$, and it can be shown that, when $c_j = 0$, Equation (62) is satisfied for all $\omega \neq d_j$ where the power is identically zero. Therefore, when $c_j \neq 0$, we require that the numerator is positive for all ω. This requirement can also be written as

$$a_j c_j > -b_j d_j \quad (63)$$

$$a_j c_j > b_j d_j. \quad (64)$$

Furthermore, we can see that a_j must be positive since $k(0) = a_j$ should be positive, and similarly, by requiring the covariance to be finite at infinite lag, we obtain the constraint $c_j \geqslant 0$. Combining these results, we find the constraint

$$|b_j d_j| < a_j c_j. \quad (65)$$

The constraint for $J > 1$ is more complicated, so in most cases we require that each term is positive definite using this relationship and use the fact that a product of sum of positive-definite kernels will also be positive definite (Rasmussen & Williams 2006) to show that the full model is positive. However, in the case of J general *celerite* terms, we can check for negative values of the PSD by solving for the roots of the power spectrum; if there are any real, positive roots, then the power spectrum goes negative (or zero) and thus does not represent a valid kernel. We rewrite the power spectrum, Equation (9)), abbreviating with $z = \omega^2$:

$$S(\omega) = \sum_{j=1}^{J} \frac{q_j z + r_j}{z^2 + s_j z + t_j} = 0, \quad (66)$$

where

$$q_j = a_j c_j - b_j d_j \quad (67)$$

$$r_j = (d_j^2 + c_j^2)(b_j d_j + a_j c_j) \quad (68)$$

$$s_j = 2(c_j^2 - d_j^2) \quad (69)$$

$$t_j = (c_j^2 + d_j^2)^2. \quad (70)$$

The denominators of each term are positive, so we can multiply through by $\prod_j (z^2 + s_j z + t_j)$ to find

$$Q_0(z) = \sum_{j=1}^{J}(q_j z + r_j)\prod_{k \neq j}(z^2 + s_k z + t_k) = 0, \quad (71)$$

which is a polynomial with order $2(J - 1) + 1$. With $J = 2$, this yields a cubic equation whose roots can be obtained exactly.

For arbitrary J, a procedure based on Sturm's theorem (Dörrie 1965) allows one to determine whether there are any real roots within the range $(0, \infty]$. We first construct $Q_0(z)$ and its derivative $Q_1(z) = Q_0'(z)$ and then loop from $k = 2$ to $k = 2(J - 1) + 1$, computing

$$Q_k(z) = -\mathrm{rem}(Q_{k-2}, Q_{k-1}), \quad (72)$$

where the function $\mathrm{rem}(p, q)$ is the remainder polynomial after dividing $p(z)$ by $q(z)$.

We evaluate the coefficients of each of the polynomials in the series by evaluating $f_0 = \{Q_0(0),\ldots,Q_{2(J-1)+1}(0)\}$ to give us the signs of these polynomials evaluated at $z = 0$. Likewise, we evaluate the coefficients of the largest-order term in each polynomial that gives the sign of the polynomial as $z \to \infty$, f_∞. With the sequence of coefficients f_0 and f_∞, we then determine how many times the sign changes in each of these, where $\sigma(0)$ is the number of sign changes at $z = 0$ and $\sigma(\infty)$ is the number of sign changes at $z \to \infty$. The total number of real roots in the range $(0, \infty]$ is given by $N_+ = \sigma(0) - \sigma(\infty)$.

We have checked that this procedure works for a wide range of parameters, and we find that it robustly matches the number of positive real roots that we evaluated numerically. The advantage of this procedure is that it does not require computing the roots, but only carrying out algebraic manipulation of polynomials to determine the number of positive real roots. If a nonzero real root is found, the likelihood may be set to zero.

Appendix B
Semiseparable Matrix Operations

In this appendix, we summarize some algorithms for manipulating semiseparable matrices and discuss how these can be used with GP models and celerite.

B.1. Multiplication

The product of a rank-R semiseparable matrix with general vectors and matrices can be computed in $\mathcal{O}(N R)$ operations (Vandebril et al. 2007). In the context of *celerite* models, naïve application of these methods will result in numerical overflow and underflow issues, much like we found with the Cholesky factorization. To avoid these errors, we derive the following numerically stable algorithm for computing the dot product of a *celerite* covariance matrix

$$\mathbf{y} = K \, \mathbf{z} \quad (73)$$

using the semiseparable representation for K defined in Equation (43):

$$f_{n,j}^+ = \phi_{n+1,j}[f_{n+1,j}^+ + \tilde{U}_{n,j} z_{n+1}] \tag{74}$$

$$f_{n,j}^- = \phi_{n,j}[f_{n-1,j}^- + \tilde{V}_{n-1,j} z_{n-1}] \tag{75}$$

$$y_n = A_{n,n} z_n + \sum_{j=1}^{2J}[\tilde{V}_{n,j}f_{n,j}^+ + \tilde{U}_{n-1,j}f_{n,j}^-], \tag{76}$$

where $f_{N,j}^+ = 0$ and $f_{1,j}^- = 0$. This algorithm requires two sweeps—$n = 2,...,N$ to compute f^- and $n = N-1,...,1$ to compute f^+—with a scaling of $\mathcal{O}(N\,J)$.

B.2. Sampling Data from a Celerite Process

To sample a data set y from a GP model for fixed parameters θ and α, we must compute

$$y = \mu_\theta(t) + K_\alpha^{1/2}\,q, \tag{77}$$

where $\mu_\theta(t)$ is the mean function evaluated at the input coordinates t, q is a vector of draws from the unit normal $q_i \sim \mathcal{N}(0, 1)$, and

$$K_\alpha = K_\alpha^{1/2}\,(K_\alpha^{1/2})^T. \tag{78}$$

For a general matrix K_α, the cost of this operation scales as $\mathcal{O}(N^3)$ to compute $K_\alpha^{1/2}$ and as $\mathcal{O}(N^2)$ to compute the dot product. For *celerite* models, we use the semiseparable representation of the Cholesky factor that we derived in Section 5.1 to compute the dot product Equation (77) in $\mathcal{O}(N\,J)$. Using the notation from Section 5.2, the algorithm to compute this dot product is

$$f_{n,j} = \phi_{n,j}[f_{n-1,j} + \tilde{W}_{n-1,j}\sqrt{D_{n-1,n-1}}\,q_{n-1}] \tag{79}$$

$$y_n = \sqrt{D_{n,n}}\,q_n + \sum_{j=1}^{2J}\tilde{U}_{n,j}f_{n,j}, \tag{80}$$

where $f_{1,j} = 0$ for all j. This scalable algorithm is useful for generating large simulated data sets for experiments with celerite and for performing posterior predictive checks.

B.3. Interpolation and Extrapolation

The predictive distribution of a GP at a vector of M input coordinates

$$y^* = (t_1^* \quad \cdots \quad t_M^*)^T, \tag{81}$$

conditioned on a data set (t, y) and GP parameters (θ, α), is a normal $y^* \sim \mathcal{N}(\mu^*, K^*)$ with mean

$$\mu^* = \mu_\theta(t^*) + K(t^*, t)K(t, t)^{-1}[y - \mu_\theta(t)] \tag{82}$$

and covariance

$$K^* = K(t^*, t^*) - K(t^*, t)K(t, t)^{-1} K(t, t^*), \tag{83}$$

where $K(v, w)$ is the covariance matrix computed between v and w (Rasmussen & Williams 2006).

Naïvely, the computational cost of computing Equation (82) for a *celerite* model scales as $\mathcal{O}(N\,M)$ if we reuse the Cholesky factor derived in Section 5.1. It is, however, possible to improve this scaling to $\mathcal{O}(n\,N + m\,M)$, where n and m are integer constants. To derive this, we expand Equation (82)

using Equation (8):

$$\mu_m^* = K(t_m^*, t)z$$
$$= \sum_{n=1}^{N}\sum_{j=1}^{J} e^{-c_j|t_m^* - t_n|}[a_j \cos(d_j\,|t_m^* - t_n|)$$
$$+ b_j \sin(d_j\,|t_m^* - t_n|)]\,z_n, \tag{84}$$

where

$$z = K(t, t)^{-1}[y - \mu_\theta(t)]. \tag{85}$$

Dividing the sum over n into $\{t_1,...,t_{n_0}\} < t_m^*$ and $\{t_{n0+1},...,t_N\} \geqslant t_m^*$ gives

$$\mu_m^* = \sum_{j=1}^{J}\sum_{n=1}^{n_0} e^{-c_j(t_m^* - t_n)}[a_j \cos(d_j(t_m^* - t_n))$$
$$+ b_j \sin(d_j(t_m^* - t_n))]\,z_n$$
$$+ \sum_{j=1}^{J}\sum_{n=n_0+1}^{N} e^{-c_j(t_n - t_m^*)}[a_j \cos(d_j(t_n - t_m^*))$$
$$+ b_j \sin(d_j(t_n - t_m^*))]\,z_n. \tag{86}$$

We compute this using two passes, a forward pass $n_0 = 1,...,N$ and a backward pass $n_0 = N,...,1$. We define

$$Q_{n,k}^- = [Q_{n-1,k}^- + z_n\,\tilde{V}_{n,k}]\,e^{-c_{k//2}(t_{n+1}-t_n)}, \tag{87}$$

$$X_{m,n,k}^- = e^{-c_{k//2}(t_m^* - t_{n+1})}\,\tilde{U}_{m,k}^*, \tag{88}$$

$$Q_{n,k}^+ = [Q_{n+1,k}^+ + z_n\,\tilde{U}_{n,k}]\,e^{-c_{k//2}(t_n-t_{n-1})}, \tag{89}$$

$$X_{m,n,k}^+ = e^{-c_{k//2}(t_{n-1}-t_m^*)}\,\tilde{V}_{m,k}^*, \tag{90}$$

where $t_0 = t_1$, $t_{N+1} = t_N$, $Q_{0,k}^- = 0$ and $Q_{N+1,k}^+ = 0$ for $k = 1$ to $2J$, and in $X_{m,n,k}^\pm$ the expressions for $\tilde{U}_{m,i}^*$ and $\tilde{V}_{m,i}^*$ are evaluated at t_m^*. For each value of m, $Q^\pm$ are recursively updated over n until n_0 is reached, at which point the predicted mean can be computed from

$$\mu_m^* = \sum_{k=1}^{2J}[Q_{n_0,k}^- X_{m,n_0,k}^- + Q_{n_0+1,k}^+ X_{m,n_0+1,k}^+]. \tag{91}$$

ORCID iDs

Daniel Foreman-Mackey ⓘ https://orcid.org/0000-0002-9328-5652
Eric Agol ⓘ https://orcid.org/0000-0002-0802-9145
Sivaram Ambikasaran ⓘ https://orcid.org/0000-0003-2978-6281
Ruth Angus ⓘ https://orcid.org/0000-0003-4540-5661

References

Agol, E., Steffen, J., Sari, R., & Clarkson, W. 2005, MNRAS, 359, 567
Aigrain, S., Llama, J., Ceillier, T., et al. 2015, MNRAS, 450, 3211
Aigrain, S., Parviainen, H., & Pope, B. J. S. 2016, MNRAS, 459, 2408
Ambikasaran, S. 2015, Numer. Linear Algebra Appl., 22, 1102
Ambikasaran, S., Foreman-Mackey, D., Greengard, L., Hogg, D. W., & O'Neil, M. 2016, ITPAM, 38, 252
Anderson, E., Bai, Z., Bischof, C., et al. 1999, LAPACK Users' Guide (3rd ed.; Philadelphia, PA: Society for Industrial and Applied Mathematics)
Anderson, E. R., Duvall, T. L., Jr., & Jefferies, S. M. 1990, ApJ, 364, 699
Angus, R., Morton, T., Aigrain, S., Foreman-Mackey, D., & Rajpaul, V. 2017, arXiv:1706.05459

Astropy Collaboration, Robitaille, T. P., Tollerud, E. J., et al. 2013, A&A, 558, A33
Barclay, T., Endl, M., Huber, D., et al. 2015, ApJ, 800, 46
Bezanzon, J., Edelman, A., Karpinski, S., & Shah, V. B. 2017, SIAM Review, 59, 65
Bond, J. R., Crittenden, R. G., Jaffe, A. H., & Knox, L. 1999, CSE, 1, 21
Bond, J. R., & Efstathiou, G. 1987, MNRAS, 226, 655
Borucki, W. J., Koch, D., Basri, G., et al. 2010, Sci, 327, 977
Brewer, B. J., Pártay, L. B., & Csányi, G. 2011, Statistics and Computing, 21, 649
Brewer, B. J., & Stello, D. 2009, MNRAS, 395, 2226
Brown, T. M. 2001, ApJ, 553, 1006
Byrd, R. H., Lu, P., Nocedal, J., & Zhu, C. 1995, SIAM Journal on Scientific Computing, 16, 1190
Campante, T. L., Schofield, M., Kuszlewicz, J. S., et al. 2016, ApJ, 830, 138
Carter, J. A., & Winn, J. N. 2009, ApJ, 704, 51
Chaplin, W. J., Basu, S., Huber, D., et al. 2013, ApJS, 210, 1
Chaplin, W. J., Kjeldsen, H., Christensen-Dalsgaard, J., et al. 2011, Sci, 332, 213
Claret, A., & Bloemen, S. 2011, A&A, 529, A75
Ivezić, Ž, Connolly, A. J., VanderPlas, J. T., & Gray, A. 2014, Statistics, Data Mining, and Machine Learning in Astronomy: A Practical Python Guide for the Analysis of Survey Data (Princeton, NJ: Princeton Univ. Press)
Corsaro, E., & Ridder, J. D. 2014, A&A, 571, A71
Czekala, I., Mandel, K. S., Andrews, S. M., et al. 2017, ApJ, 840, 49
Dawson, R. I., Johnson, J. A., Fabrycky, D. C., et al. 2014, ApJ, 791, 89
Dillon, J. S., Liu, A., & Tegmark, M. 2013, PhRvD, 87, 043005
Dörrie, H. 1965, 100 Great Problems of Elementary Mathematics: Their History and Solution, Dover Books on Mathematics Series, Section 24 (New York: Dover)
Dumusque, X., Boisse, I., & Santos, N. C. 2014, ApJ, 796, 132
Evans, T. M., Aigrain, S., Gibson, N., et al. 2015, MNRAS, 451, 680
Feroz, F., Hobson, M. P., & Bridges, M. 2009, MNRAS, 398, 1601
Foreman-Mackey, D. 2016, JOSS, 24 doi:10.21105/joss.00024
Foreman-Mackey, D., Hogg, D. W., Lang, D., & Goodman, J. 2013, PASP, 125, 306
Foreman-Mackey, D., Hoyer, S., Bernhard, J., & Angus, R. 2014, George: George (v0.2.0), Zenodo, doi:10.5281/zenodo.11989
Foreman-Mackey, D., & Morton, T. 2016, dfm/transit: v0.3.0, Zenodo, doi:10.5281/zenodo.159478
Foreman-Mackey, D., Morton, T. D., Hogg, D. W., Agol, E., & Schölkopf, B. 2016, AJ, 152, 206
Gibson, N. P., Aigrain, S., Roberts, S., et al. 2012, MNRAS, 419, 2683
Gilliland, R. L., Brown, T. M., Christensen-Dalsgaard, J., et al. 2010, PASP, 122, 131
Goodman, J., & Weare, J. 2010, Communications in Applied Mathematics and Computational Science, 5, 65
Gould, A., Huber, D., Penny, M., & Stello, D. 2015, JKAS, 48, 93
Gregory, P. 2005, Bayesian Logical Data Analysis for the Physical Sciences: A Comparative Approach with Mathematica® Support (Cambridge: Cambridge Univ. Press)
Grunblatt, S. K., Huber, D., Gaidos, E. J., et al. 2016, AJ, 152, 185
Guennebaud, G., Jacob, B., et al. 2010, Eigen v3, http://eigen.tuxfamily.org
Harvey, J. 1985, in ASP Conf. Ser. 235, ESA Future Missions in Solar, Heliospheric & Space Plasma Physics, ed. E. Rolfe & B. Battrick (San Francisco, CA: ASP), 199
Haywood, R. D., Cameron, A. C., Queloz, D., et al. 2014, MNRAS, 443, 2517
Holman, M. J. 2005, Sci, 307, 1288
Howell, S. B., Sobeck, C., Haas, M., et al. 2014, PASP, 126, 398
Huber, D., Bedding, T. R., Stello, D., et al. 2011, ApJ, 743, 143
Huber, D., Stello, D., Bedding, T. R., et al. 2009, CoAst, 160, 74
Hunter, J. D. 2007, CSE, 9, 90
Ivezić, V., Tyson, J. A., Acosta, E., et al. 2008, arXiv:0805.2366v4
Jones, E., Oliphant, T., Peterson, P., et al. 2001, SciPy: Open Source Scientific Tools for Python, https://www.scipy.org/citing.html

Kallinger, T., De Ridder, J., Hekker, S., et al. 2014, A&A, 570, A41
Kelly, B. C., Becker, A. C., Sobolewska, M., Siemiginowska, A., & Uttley, P. 2014, ApJ, 788, 33
Kelly, B. C., Sobolewska, M., & Siemiginowska, A. 2011, ApJ, 730, 52
Kipping, D. M. 2013, MNRAS, 435, 2152
Knutson, H. A., Charbonneau, D., Allen, L. E., et al. 2007, Natur, 447, 183
Littlefair, S. P., Burningham, B., & Helling, C. 2017, MNRAS, 466, 4250
Liu, J. S. 2008, Monte Carlo Strategies in Scientific Computing (New York: Springer-Verlag)
Loeb, A., & Gaudi, B. S. 2003, ApJ, 588, L117
Luger, R., Agol, E., Kruse, E., et al. 2016, AJ, 152, 100
Luger, R., Kruse, E., Foreman-Mackey, D., Agol, E., & Saunders, N. 2017, arXiv:1702.05488
MacLeod, C. L., Ivezić, Ž, Kochanek, C. S., et al. 2010, ApJ, 721, 1014
Mandel, K., & Agol, E. 2002, ApJ, 580, L171
Mathur, S., García, R. A., Ballot, J., et al. 2014, A&A, 562, A124
McAllister, M. J., Littlefair, S. P., Dhillon, V. S., et al. 2016, MNRAS, 464, 1353
McQuillan, A., Aigrain, S., & Mazeh, T. 2013, MNRAS, 432, 1203
McQuillan, A., Mazeh, T., & Aigrain, S. 2014, ApJS, 211, 24
Michel, E., Samadi, R., Baudin, F., et al. 2009, A&A, 495, 979
Nocedal, J., & Wright, S. J. 2006, Numerical Optimization (Berlin: Springer)
Pancoast, A., Brewer, B. J., Treu, T., et al. 2014, MNRAS, 445, 3073
Pinsonneault, M. H., Elsworth, Y., Epstein, C., et al. 2014, ApJS, 215, 19
Pope, B. J. S., Parviainen, H., & Aigrain, S. 2016, MNRAS, 461, 3399
Press, W. H., & Rybicki, G. B. 1998, ApJ, 507, 108
Rajpaul, V., Aigrain, S., Osborne, M. A., Reece, S., & Roberts, S. 2015, MNRAS, 452, 2269
Rasmussen, C. E., & Williams, K. I. 2006, Gaussian Processes for Machine Learning (Cambridge, MA: MIT Press)
Rauer, H., Catala, C., Aerts, C., et al. 2014, ExA, 38, 249
Ricker, G. R., Winn, J. N., Vanderspek, R., et al. 2014, Proc. SPIE, 9143, 914320
Rybicki, G. B., & Press, W. H. 1992, ApJ, 398, 169
Rybicki, G. B., & Press, W. H. 1995, PhRvL, 74, 1060
Schwarz, G. 1978, AnSta, 6, 461
Smith, J. C., Stumpe, M. C., Van Cleve, J. E., et al. 2012, PASP, 124, 1000
Sokal, A. D. 1989, Monte Carlo Methods in Statistical Mechanics: Foundations and New Algorithms, Troisieme Cycle de la Physique en Suisse Romande (New York: Springer)
Spergel, D., Gehrels, N., Baltay, C., et al. 2015, arXiv:1503.03757
Stello, D., Cantiello, M., Fuller, J., et al. 2016, Natur, 529, 364
Stello, D., Chaplin, W. J., Basu, S., Elsworth, Y., & Bedding, T. R. 2009, MNRAS, 400, L80
Stello, D., Huber, D., Bedding, T. R., et al. 2013, ApJ, 765, L41
Stumpe, M. C., Smith, J. C., Van Cleve, J. E., et al. 2012, PASP, 124, 985
Uhlenbeck, G. E., & Ornstein, L. S. 1930, PhRv, 36, 823
Uttley, P., McHardy, I. M., & Vaughan, S. 2005, MNRAS, 359, 345
Vandebril, R., Van Barel, M., & Mastronardi, N. 2007, Matrix Computations and Semiseparable Matrices: Linear Systems, Vol 1 (Baltimore, MD: Johns Hopkins Univ. Press)
VanderPlas, J. T. 2017, arXiv:1703.09824
Van Der Walt, S., Colbert, S. C., & Varoquaux, G. 2011, CSE, 13, 22
Verner, G. A., Elsworth, Y., Chaplin, W. J., et al. 2011, MNRAS, 415, 3539
Wandelt, B. D., & Hansen, F. K. 2003, PhRvD, 67, 023001
Wang, Y., Khardon, R., & Protopapas, P. 2012, ApJ, 756, 67
Wilson, A. G. 2014, PhD thesis, Univ. Cambridge
Wilson, A. G., & Adams, R. P. 2013, In ICML, 1067
Wilson, A. G., Dann, C., & Nickisch, H. 2015, arXiv:1511.01870
Wilson, A. G., & Nicisch, H. 2015, in Proc. 32nd Int. Conf. on Machine Learning, ed. F. Bach & D. Blei, 1775
Zhu, C., Byrd, R. H., Lu, P., & Nocedal, J. 1997, ACM Transactions on Mathematical Software (TOMS), 23, 550
Zinn, J. C., Kochanek, C. S., Kozłowski, S., et al. 2016, arXiv:1612.04834
Zu, Y., Kochanek, C. S., & Peterson, B. M. 2011, ApJ, 735, 80
Zucker, S., Mazeh, T., & Alexander, T. 2007, ApJ, 670, 1326