Draft version June 5, 2017
Preprint typeset using LaTeX style AASTeX6 with modifications

FAST AND SCALABLE GAUSSIAN PROCESS MODELING WITH APPLICATIONS TO ASTRONOMICAL TIME SERIES

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

[1]Sagan Fellow
[2]Astronomy Department, University of Washington, Seattle, WA
[3]Guggenheim Fellow
[4]Simons Fellow
[5]Department of Astronomy, Columbia University, New York, NY
[6]Department of Computational and Data Sciences, Indian Institute of Science, Bangalore, India

ABSTRACT

The growing field of large-scale time domain astronomy requires methods for probabilistic data analysis that are computationally tractable, even with large datasets. Gaussian Processes are a popular class of models used for this purpose but, since the computational cost scales as the cube of the number of data points, their application has been limited to relatively small datasets. In this paper, we present a method for Gaussian Process modeling in one-dimension where the computational requirements scale linearly with the size of the dataset. We demonstrate the method by applying it to simulated and real astronomical time series datasets. 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 is effective in other cases. We present a mathematical description of the method, the details of the implementation, and a comparison to existing scalable Gaussian Process methods. The method is flexible, fast, and most importantly, interpretable, with a wide 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.

Keywords: methods: data analysis — methods: statistical — asteroseismology — stars: rotation — planetary systems

1. INTRODUCTION

DFM: Talk about LSST here too!

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 (Czekala et al. 2017; Evans et al. 2015). While these models are widely applicable, their use has been limited, in practice, by the computational cost and scaling. In general, the cost of computing a 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 $\sim 10^{4-9}$ targets with $\sim 10^{3-5}$ observations each – this scaling is prohibitive.

In this paper, we present a method for computing a class of GP models that scales linearly with the number of data points $\mathcal{O}(N)$ for one dimensional data sets. *DFM*: This method is a generalization of a method developed by Ambikasaran (2015) (**This is no longer correct -different Ambikasaran result!**) that was, in turn, built on intuition from a twenty year old paper (Rybicki & Press 1995). This method can only be used with one-dimensional datasets 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, there is no further constraint on the data or the model. In particular, the measurements don't 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. *DFM*: More precisely, the matrix can be embedded in an equivalent higher-dimensional system that can be solved efficiently using a sparse or banded solver. Replace with(?): The matrix can be expressed as the sum of semi-separable matrices, each of which is constructed from the outer product of one-dimensional vectors. These are a type of low-rank matrix, which may be solved by Cholesky decomposition using a novel technique developed by one of us (SA). As we will discuss in the following pages, this choice of kernel function is not as restrictive as it first seems because it arises naturally in some physical systems and it can be used as an effective model in many other cases. This method is especially appealing compared to other similar methods – we return to these below – because it is exact, flexible, 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 the method described

in the following pages will benefit this field but we also expect that there will be applications in other branches of astrophysics and beyond. In Section **??**, we present applications of the method to research problems in stellar rotation, asteroseismic analysis, and exoplanet transit fitting. Some readers might consider starting with that section for some motivating examples before delving into the detailed derivations of the earlier sections.

In the following pages, we motivate the general problem of GP modeling, describe the *DFM*: (delete) previously published scalable method (Rybicki & Press 1995; Ambikasaran 2015) and our generalization, and demonstrate the model's application on various 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 (Foreman-Mackey et al. 2017).

2. GAUSSIAN PROCESSES

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

$$\boldsymbol{y} = \begin{pmatrix} y_1 & \cdots & y_N \end{pmatrix}^{\mathrm{T}} \tag{1}$$

at coordinates

$$X = \begin{pmatrix} \boldsymbol{x}_1 & \cdots & \boldsymbol{x}_N \end{pmatrix}^{\mathrm{T}} \tag{2}$$

is

$$\ln \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\alpha}) = \ln p(\boldsymbol{y} \,|\, X, \boldsymbol{\theta}, \boldsymbol{\alpha}) = -\frac{1}{2} \boldsymbol{r}_{\boldsymbol{\theta}}{}^{\mathrm{T}} \, K_{\boldsymbol{\alpha}}{}^{-1} \, \boldsymbol{r}_{\boldsymbol{\theta}} - \frac{1}{2} \ln \det K_{\boldsymbol{\alpha}} - \frac{N}{2} \ln (2\pi) \tag{3}$$

where

$$\boldsymbol{r}_{\boldsymbol{\theta}} = \begin{pmatrix} y_1 - \mu_{\boldsymbol{\theta}}(\boldsymbol{x}_1) & \cdots & y_N - \mu_{\boldsymbol{\theta}}(\boldsymbol{x}_N) \end{pmatrix}^{\mathrm{T}} \tag{4}$$

is the vector of residuals and the elements of the covariance matrix K are given by $[K_{\boldsymbol{\alpha}}]_{nm} = k_{\boldsymbol{\alpha}}(\boldsymbol{x}_n, \boldsymbol{x}_m)$. 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. The maximum likelihood values for the parameters $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}$ for a given dataset $(\boldsymbol{y}, X)$ can be found by maximizing Equation (3) with respect to $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}$ using a non-linear optimization routine (Nocedal & Wright 2006). Similarly, probabilistic constraints on $\boldsymbol{\theta}$ and $\boldsymbol{\alpha}$ can be obtained by multiplying the likelihood by a prior $p(\boldsymbol{\theta}, \boldsymbol{\alpha})$ 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 datasets because the cost of computing the inverse and determinant of the matrix $K_{\boldsymbol{\alpha}}$ is $\mathcal{O}(N^3)$. In other words, this cost is proportional to

the cube of the number of data points N. This means that for large datasets, every evaluation of the likelihood quickly becomes computationally intractable. In this case, the use of non-linear optimization or MCMC is no longer practical.

In the following section, we present a method for substantially improving this scaling in many circumstances. We call our method and its implementations celerite.[1] The celerite method requires using a specific model for the covariance $k_{\boldsymbol{\alpha}}(\boldsymbol{x}_n, \boldsymbol{x}_m)$ 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 datasets, where by "one-dimensional" we mean that the *input coordinates* $\boldsymbol{x}_n$ are scalar, $\boldsymbol{x}_n \equiv t_n$. We use t as the input coordinate because one-dimensional GPs are often applied to time series data but this isn't a real restriction and the celerite method can be applied to *any* one-dimensional dataset. Furthermore, the covariance function for the celerite method is "stationary". In other words, $k_{\boldsymbol{\alpha}}(t_n, t_m)$ is only a function of $\tau_{nm} \equiv |t_n - t_m|$.

3. THE CELERITE MODEL

To scale GP models to larger datasets, 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

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

where $\{{\sigma_n}^2\}_{n=1}^N$ are the measurement uncertainties, δ_{nm} is the Kronecker delta, and $\boldsymbol{\alpha} = (a, c)$. The intuition behind this method is that, for this choice of $k_{\boldsymbol{\alpha}}$, the inverse of $K_{\boldsymbol{\alpha}}$ is tridiagonal and can be computed with a small number of operations for each data point. Subsequently, Ambikasaran (2015) generalized this method to arbitrary mixtures of exponentials

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

In this case, the inverse is dense but Equation (3) can still be evaluated in $\mathcal{O}(N\,J^2)$, where J is the number of components in the mixture.

This kernel function can be made even more general by introducing complex parameters $a_j \to a_j \pm i\,b_j$ and $c_j \to c_j \pm i\,d_j$. In this case, the covariance function

[1] The name celerite comes from the French word *célerité* 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 sans-serif.

becomes

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

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 the following section 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 the autocorrelation structure is defined by a mixture of quasiperiodic oscillators

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

For clarity, we refer to the argument within the sum as a "*celerite* term" for the remainder of this paper. The Fourier transform[2] of this covariance function is the power spectral density (PSD) of the process and it is given by

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

The physical interpretation of this model isn't 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

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

and the PSD for this component is

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

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 quasiperiodic 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

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

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

with the PSD

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

This model is often called an Ornstein–Uhlenbeck process (in reference to the classic paper, Uhlenbeck & Ornstein 1930) and this is the kernel that was studied by Rybicki & Press (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 re-written as a sum with updated parameters

$$k_j(\tau)\,k_k(\tau) = e^{-\tilde{c}\,\tau}\,[\tilde{a}_+\,\cos(\tilde{d}_+\,\tau) + \tilde{b}_+\,\sin(\tilde{d}_+\,\tau) + \tilde{a}_-\,\cos(\tilde{d}_-\,\tau) + \tilde{b}_-\,\sin(\tilde{d}_-\,\tau)] \tag{14}$$

where

$$\begin{aligned}
\tilde{a}_\pm &= \frac{1}{2}\,(a_j\,a_k \pm b_j\,b_k) && (15)\\
\tilde{b}_\pm &= \frac{1}{2}\,(b_j\,a_k \mp a_j\,b_k) && (16)\\
\tilde{c} &= c_j + c_k && (17)\\
\tilde{d}_\pm &= d_j \mp d_k \quad . && (18)
\end{aligned}$$

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

4. SEMI-SEPARABLE MATRIX FACTORIZATION

The relationship between the *celerite* model and semi-separable linear algebra was first recognized by Ambikasaran (Ambikasaran 2015). A rank-R semi-separable matrix is a matrix K where the elements can be written as

$$K_{n,m} = \begin{cases} \sum_{r=1}^{R} U_{n,r}\,V_{m,r} & \text{if } m < n \\ A_{n,n} & \text{if } m = n \\ \sum_{r=1}^{R} U_{m,r}\,V_{n,r} & \text{otherwise} \end{cases} \tag{19}$$

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

$$K = A + \mathrm{tril}(U\,V^{\mathrm{T}}) + \mathrm{triu}(V\,U^{\mathrm{T}}) \tag{20}$$

where the tril function extracts the strict lower triangular component of its argument and triu is the equivalent strict upper triangular operation. Note that the last two terms in this equation are transposes of one another. For the *celerite* model described in the previous section, the rank of the semi-separable matrix is $R = 2\,J$ and the

components of the relevant matrices are

$$
\begin{aligned}
U_{n,2\,j-1} &= a_j\, e^{-c_j\, t_n}\, \cos(d_j\, t_n) + b_j\, e^{-c_j\, t_n}\, \sin(d_j\, t_n) && (21)\\
U_{n,2\,j} &= a_j\, e^{-c_j\, t_n}\, \sin(d_j\, t_n) - b_j\, e^{-c_j\, t_n}\, \cos(d_j\, t_n) && (22)\\
V_{m,2\,j-1} &= e^{c_j\, t_m}\, \cos(d_j\, t_m) && (23)\\
V_{m,2\,j} &= e^{c_j\, t_m}\, \sin(d_j\, t_m) && (24)\\
A_{n,n} &= \sigma_n^2 + \sum_{j=1}^{J} a_j \quad . && (25)
\end{aligned}
$$

To help clarify these equations, we write out here an example of a semi-separable matrix with a single *celerite* term with $c = d = 1$:

$$
\begin{aligned}
K &= \begin{pmatrix}
a+{\sigma_1}^2 & e^{-\tau_{21}}(a\,\cos\tau_{21}+b\,\sin\tau_{21}) & e^{-\tau_{31}}(a\,\cos\tau_{31}+b\,\sin\tau_{31}) & \dots \\
e^{-\tau_{21}}(a\,\cos\tau_{21}+b\,\sin\tau_{21}) & a+{\sigma_2}^2 & e^{-\tau_{32}}(a\,\cos\tau_{32}+b\,\sin\tau_{32}) & \dots \\
e^{-\tau_{32}}(a\,\cos\tau_{32}+b\,\sin\tau_{32}) & e^{-\tau_{32}}(a\,\cos\tau_{32}+b\,\sin\tau_{32}) & a+{\sigma_3}^2 & \dots \\
\dots & \dots & \dots & \dots
\end{pmatrix} && (26)\\
&= \begin{pmatrix}
a+{\sigma_1}^2 & 0 & 0 & \dots \\
0 & a+{\sigma_2}^2 & 0 & \dots \\
0 & 0 & a+{\sigma_3}^2 & \dots \\
\dots & \dots & \dots & \dots
\end{pmatrix} && (27)\\
&+ \operatorname{tril}\left[
\begin{pmatrix}
e^{-t_1}(a\,\cos t_1+b\,\sin t_1) & e^{-t_1}(a\,\sin t_1-b\,\cos t_1) \\
e^{-t_2}(a\,\cos t_2+b\,\sin t_2) & e^{-t_2}(a\,\sin t_2-b\,\cos t_2) \\
e^{-t_3}(a\,\cos t_3+b\,\sin t_3) & e^{-t_3}(a\,\sin t_3-b\,\cos t_3) \\
\dots & \dots
\end{pmatrix} \times
\begin{pmatrix}
e^{t_1}\cos t_1 & e^{t_1}\sin t_1 \\
e^{t_2}\cos t_2 & e^{t_2}\sin t_2 \\
e^{t_3}\cos t_3 & e^{t_3}\sin t_3 \\
\dots & \dots
\end{pmatrix}^T
\right] && (28)\\
&+ \operatorname{triu}\left[
\begin{pmatrix}
e^{t_1}\cos t_1 & e^{t_1}\sin t_1 \\
e^{t_2}\cos t_2 & e^{t_2}\sin t_2 \\
e^{t_3}\cos t_3 & e^{t_3}\sin t_3 \\
\dots & \dots
\end{pmatrix} \times
\begin{pmatrix}
e^{-t_1}(a\,\cos t_1+b\,\sin t_1) & e^{-t_1}(a\,\sin t_1-b\,\cos t_1) \\
e^{-t_2}(a\,\cos t_2+b\,\sin t_2) & e^{-t_2}(a\,\sin t_2-b\,\cos t_2) \\
e^{-t_3}(a\,\cos t_3+b\,\sin t_3) & e^{-t_3}(a\,\sin t_3-b\,\cos t_3) \\
\dots & \dots
\end{pmatrix}^T
\right] && (29)\\
&= A + \operatorname{tril}(UV^T) + \operatorname{triu}(VU^T). && (30)
\end{aligned}
$$

Note that rather than storing the full $N \times N$ covariance matrix, it is only necessary to store the $2J \times N$ matrices U and V, as well as the diagonal terms, for a total storage volume of $(4J + 1) \times N$ floating point numbers.

4.1. *Cholesky factorization*

To compute the marginalized likelihood defined in Equation (3), we must compute the log-determinant of K and solve linear systems of the form $K\,\boldsymbol{z} = \boldsymbol{y}$ for $\boldsymbol{z}$, where K is a rank-$2\,J$ semi-separable matrix. If K is positive semi-definite, these can be computed using the Cholesky factorization of K:

$$K = L\,D\,L^{\mathrm{T}} \tag{31}$$

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

$$L = I + \mathrm{tril}(U\,W^T) \tag{32}$$

where I is the identity, U is the matrix from above, and W is an unknown $N \times 2\,J$ matrix. Combining this with Equation (20), we find the equation

$$\begin{aligned} A + \mathrm{tril}(U\,V^{\mathrm{T}}) + \mathrm{triu}(V\,U^{\mathrm{T}}) &= \left[I + \mathrm{tril}(U\,W^T)\right]\,D\,\left[I + \mathrm{tril}(U\,W^T)\right]^{\mathrm{T}} \\ &= D + \mathrm{tril}(U\,W^{\mathrm{T}})\,D + D\,\mathrm{triu}(W\,U^{\mathrm{T}}) \\ &\quad + \mathrm{tril}(U\,W^{\mathrm{T}})\,D\,\mathrm{triu}(W\,U^{\mathrm{T}}) \end{aligned} \tag{33}$$

that we can solve for W. Setting each element on the left side of Equation (33) equal to the corresponding element on the right side, we derive the following recursion relations

$$\begin{aligned} S_{n,j,k} &= S_{n-1,j,k} + D_{n-1,n-1}\,W_{n-1,j}\,W_{n-1,k} \\ D_{n,n} &= A_{n,n} - \sum_{j=1}^{2\,J}\sum_{k=1}^{2\,J} U_{n,j}\,S_{n,j,k}\,U_{n,k} \\ W_{n,j} &= \frac{1}{D_{n,n}}\left[V_{n,j} - \sum_{k=1}^{2\,J} U_{n,k}\,S_{n,j,k}\right] \end{aligned} \tag{34}$$

where S_n is a symmetric $2\,J \times 2\,J$ matrix and every element $S_{1,j,k} = 0$. The computational cost of one iteration of the update in Equation (34) is $\mathcal{O}(J^2)$ and this must be run N times – once for each observation – so the total cost of this factorization scales as $\mathcal{O}(N\,J^2)$.

Note that if some *celerite* terms are real, with $b_j = d_j = 0$, then these terms only add one to the rank of U and V. We use this fact to simplify the algebra when real components are included: if J_{real} and $J_{complex}$ terms are included, with $J_{real} + J_{complex} = J$, then the U, V, and W matrices have size $N \times (J_{real} + 2J_{complex})$.

4.2. *Pre-conditioning*

Naïve implementation of the algorithm in Equation (34) will have 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 can be avoided with

the appropriate reparameterization. We define the following pre-conditioned variables

$$\begin{aligned}
\tilde{U}_{n,2\,j-1} &= a_j\,\cos(d_j\,t_n) + b_j\,\sin(d_j\,t_n) && (35)\\
\tilde{U}_{n,2\,j} &= a_j\,\sin(d_j\,t_n) - b_j\,\cos(d_j\,t_n) && (36)\\
\tilde{V}_{m,2\,j-1} &= \cos(d_j\,t_m) && (37)\\
\tilde{V}_{m,2\,j} &= \sin(d_j\,t_m) && (38)\\
\tilde{W}_{n,2\,j-1} &= e^{-c_j\,t_n}\,W_{n,2\,j-1} && (39)\\
\tilde{W}_{n,2\,j} &= e^{-c_j\,t_n}\,W_{n,2\,j} && (40)
\end{aligned}$$

and the set of variables

$$\phi_{n,2\,j-1} = \phi_{n,2\,j} = e^{-c_j\,(t_n - t_{n-1})} \tag{41}$$

with the constraint

$$\phi_{1,2\,j-1} = \phi_{1,2\,j} = 0 \quad . \tag{42}$$

Using this parameterization, we find the following, numerically stable, algorithm

$$\begin{aligned}
S_{n,j,k} &= \phi_{n,j}\,\phi_{n,k}\,\left[S_{n-1,j,k} + D_{n-1,n-1}\,\tilde{W}_{n-1,j}\,\tilde{W}_{n-1,k}\right]\\
D_{n,n} &= A_{n,n} - \sum_{j=1}^{2\,J}\sum_{k=1}^{2\,J}\tilde{U}_{n,j}\,S_{n,j,k}\,\tilde{U}_{n,k}\\
\tilde{W}_{n,j} &= \frac{1}{D_{n,n}}\left[\tilde{V}_{n,j} - \sum_{k=1}^{2\,J}\tilde{U}_{n,k}\,S_{n,j,k}\right] \quad . && (43)
\end{aligned}$$

4.3. *Solving*

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

$$\ln\det K = \sum_{n=1}^{N}\ln D_{n,n} \tag{44}$$

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

The inverse of K can be applied using the relationship

$$\boldsymbol{z} = K^{-1}\,\boldsymbol{y} = \left(L^{\mathrm{T}}\right)^{-1}\,D^{-1}\,L^{-1}\,\boldsymbol{y} \quad . \tag{45}$$

First, we solve for $\boldsymbol{z}' = L^{-1}\,\boldsymbol{y}$ using the following forward substitution algorithm

$$\begin{aligned}
f_{n,j} &= \phi_{n,j}\,\left[f_{n-1,j} + \tilde{W}_{n-1,j}\,z'_{n-1}\right]\\
z'_n &= y_n - \sum_{j=1}^{2\,J}\tilde{U}_{n,j}\,f_{n,j} && (46)
\end{aligned}$$

where $f_{0,j} = 0$ for all j. Using this result and the following backward substitution algorithm, we can compute $\boldsymbol{z}$

$$\begin{aligned} g_{n,j} &= \phi_{n+1,j}\left[g_{n+1,j} + \tilde{U}_{n+1,j}\, z_{n+1}\right] \\ z_n &= \frac{z'_n}{D_{n,n}} - \sum_{j=1}^{2\,J} \tilde{W}_{n,j}\, g_{n,j} \end{aligned} \tag{47}$$

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

4.4. *Simulation*

Simulation of a Gaussian process for large N requires multiplying an N-dimensional vector of Gaussian normal deviates, $\boldsymbol{q}$, by the square root of the covariance matrix:

$$\boldsymbol{y} = K_{\boldsymbol{\alpha}}^{1/2}\boldsymbol{q}, \tag{48}$$

where $q_i \sim N(0,1)$. Note that $K_{\boldsymbol{\alpha}} = K_{\boldsymbol{\alpha}}^{1/2}\left(K_{\boldsymbol{\alpha}}^{1/2}\right)^T$, which comparing with the Cholesky factorization gives $K_{\boldsymbol{\alpha}}^{1/2} = LD^{1/2}$, where $(D^{1/2})_{ii} = \sqrt{D_{ii}}$. Also note that $\mathcal{Q} = -2\mathcal{L}$ with $\mu_{\boldsymbol{\theta}} = 0$ is given by

$$\begin{aligned} \mathcal{Q} &= \boldsymbol{y}^T K_{\boldsymbol{\alpha}}^{-1}\boldsymbol{y} && (49)\\ &= \boldsymbol{y}^T\left(K_{\boldsymbol{\alpha}}^{-1/2}\right)^T K_{\boldsymbol{\alpha}}^{-1/2}\boldsymbol{y} && (50)\\ &= \boldsymbol{q}^T K_{\boldsymbol{\alpha}}^{1/2}\left(K_{\boldsymbol{\alpha}}^{-1/2}\right)^T K_{\boldsymbol{\alpha}}^{-1/2} K_{\boldsymbol{\alpha}}^{1/2}\boldsymbol{q} && (51)\\ &= \boldsymbol{q}^T\boldsymbol{q}. && (52) \end{aligned}$$

The last quantity is simply the dot product of an N-dimensional vector of independent standard normal random variables with itself, which is distributed according to the chi-squared distribution with N degrees of freedom, i.e. $\mathcal{Q} \sim \chi^2_N$.

To generate a draw from a Gaussian process with a specified celerite model requires carrying out the low-rank Cholesky decomposition first, generating the vector $\boldsymbol{q}$ of random normal deviates, and then multiplying

$$\boldsymbol{y} = LD^{1/2}\boldsymbol{q}. \tag{53}$$

This may be carried out in $\mathcal{O}(JN)$ operations with

$$\begin{aligned} f_{n,j} &= \phi_{n-1,j}\left(f_{n-1,j} + \tilde{W}_{n-1,j} D^{1/2}_{n-1,n-1} q_{n-1}\right) && (54)\\ y_n &= D^{1/2}_{n,n} q_n + \sum_{k=1}^{2J} \tilde{U}_{n,k} f_{n,k} && (55) \end{aligned}$$

for $n = 1$ to N and $j = 1$ to $2J$, with $f_{0,j} = 0$ for all j and $q_0 = 0$. *Agol*: Check that these equations are in fact correct. We expect that this simulation algorithm will be useful for test-driving Gaussian process inference algorithms applied to large datasets;

even simulation of Gaussian processes becomes prohibitive for large N if standard Cholesky decomposition and multiplication are used.

4.5. *Multiplication and filtering*

Multiplication of the full covariance matrix times a vector can be accomplished in $\mathcal{O}(JN)$ operations with a stable recursive algorithm using the semi-separable form:

$$\boldsymbol{y} = \left(A + \mathrm{tril}(UV^T) + \mathrm{triu}(VU^T)\right)\boldsymbol{z}. \tag{56}$$

The algorithm for multiplication is given by

$$f^+_{n,j} = \phi_{n+1,j}\left(f^+_{n+1,j} + \sum_{j=1}^{J}\tilde{U}_{n+1,j}z_{n+1}\right) \tag{57}$$

$$f^-_{n,j} = \phi_{n,j}\left(f^-_{n-1,j} + \sum_{j=1}^{J}\tilde{V}_{n-1,j}z_{n-1}\right) \tag{58}$$

$$y_n = A_{n,n}z_n + \sum_{j=1}^{J}\left[\tilde{V}_{n,j}f^+_{n,j} + \tilde{U}_{n,j}f^-_{n,j}\right], \tag{59}$$

where $f^+_{N,j} = 0$ and $f^-_{1,j} = 0$ for all $j = 1, ..., J$. The recursion requires two sweeps, a sweep downward from $n = N-1$ to $n = 1$ to compute $f^+_{n,j}$ and a sweep upwards from $n = 2$ to $n = N$ to compute $f^-_{n,j}$. *DFM*: I'm getting confused by zero-based indexing in your C code... this may be worth scrutinizing to see if I screwed up.

Multiplication of the full covariance matrix is useful, for example, for isolating a particular component of a kernel, which amounts to a customized filter of a dataset. For example, a white-noise filter is given by:

$$\bar{f} = K(\boldsymbol{t},\boldsymbol{t})\left[K(\boldsymbol{t},\boldsymbol{t}) + \sigma^2 I\right]^{-1}\boldsymbol{y}, \tag{60}$$

$$= K(\boldsymbol{t},\boldsymbol{t})\boldsymbol{z} \tag{61}$$

where $\boldsymbol{y}$ is the vector of measurements and $\boldsymbol{z}$ may be computed from the Cholesky decomposition as described above. The second step multiplication of the white-noise-free kernel times $\boldsymbol{z}$ may be accomplished in $\mathcal{O}(JN)$ steps with this multiplication algorithm. Note that for this particular example it is more efficient to compute $\bar{f} = \boldsymbol{y} - \sigma^2 I\boldsymbol{z}$, but in general the filtering kernel will not differ from the full kernel by a diagonal matrix.

4.6. *Prediction and interpolation*

For predicting or interpolating missing or future data points (and their variance) at times $\boldsymbol{t^*} = \{t^*_1, t^*_2, ..., t^*_M\}$, we need to apply the covariance between the measured and predicted times. The mean of the predicted values is (equation 2.19 from Rasmussen

& Williams):

$$\bar{f}_* = K(\boldsymbol{t}^*, \boldsymbol{t}) \left[K(\boldsymbol{t}, \boldsymbol{t}) + \sigma^2 I\right]^{-1} \boldsymbol{y}, \quad (62)$$

$$= K(\boldsymbol{t}^*, \boldsymbol{t}) \boldsymbol{z} \quad (63)$$

where $\boldsymbol{y}$ is the vector of measurements and σ is the measurement uncertainty, and $\boldsymbol{z} = \left[K(\boldsymbol{t}, \boldsymbol{t}) + \sigma^2 I\right]^{-1} \boldsymbol{y}$ can be computed from the Cholesky decomposition of the full covariance matrix in $\mathcal{O}(J^2 N)$.

Since the covariance matrix may be written as the sum of exponentials, the predicted mean is given by

$$\bar{f}^*_m = K(t^*_m, \boldsymbol{t}) \boldsymbol{z} \quad (64)$$

$$= \sum_n \sum_{j=1}^{J} e^{-c_j |t^*_m - t_n|} \left(a_j \cos\left(d_j |t^*_m - t_n|\right) + b_j \sin\left(d_j |t^*_m - t_n|\right)\right) z_j, \quad (65)$$

for $m = 1$ to M. Dividing the sum over n into $\{t_1, ..., t_{n_0}\} < t^*_m$ and $\{t_{n0+1}, ..., t_N\} \geq t^*_m$, gives

$$\bar{f}^*_m = \sum_{j=1}^{J} \sum_{n=1}^{n_0} e^{-c_j (t^*_m - t_n)} \left[a_j \cos\left(d_j (t^*_m - t_n)\right) + b_j \sin\left(d_j (t^*_m - t_n)\right)\right] z_n \quad (66)$$

$$+ \sum_{j=1}^{J} \sum_{n=n_0+1}^{N} e^{-c_j (t_n - t^*_m)} \left[a_j \cos\left(d_j (t_n - t^*_m)\right) + b_j \sin\left(d_j (t_n - t^*_m)\right)\right] z_n. \quad (67)$$

We can compute this recursively. Defining

$$Q^-_{n,i} = \left(Q^-_{n-1,i} + z_n \tilde{V}_{n,i}\right) e^{-c_j (t_{n+1} - t_n)}, \quad (68)$$

$$X^-_{m,n,i} = e^{-c_j (t^*_m - t_{n+1})} \tilde{U}_{m,i}, \quad (69)$$

$$Q^+_{n,i} = \left(Q^+_{n+1,i} + z_n \tilde{U}_{n,i}\right) e^{-c_j (t_n - t_{n-1})}, \quad (70)$$

$$X^+_{m,n,i} = e^{-c_j (t_{n-1} - t^*_m)} \tilde{V}_{m,i}, \quad (71)$$

where $t_0 = t_1$, $t_{N+1} = t_N$, $Q^-_{0,i} = 0$ and $Q^+_{N+1,i} = 0$ for $i = 1$ to $2J$, and in $X^{\pm}_{m,n,i}$ the expressions for $\tilde{U}_{m,i}$ and $\tilde{V}_{m,i}$ are evaluted at t^*_m. Two sweeps must be carried out, upward in n and m to compute Q^- recursively, and downward in m and n to compute Q^+ recursively, for a total scaling of $\approx \mathcal{O}(J^2 N + J(N + M))$. 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

$$\bar{f}^*_i = \sum_{i=1}^{2J} \left[Q^-_{n_0,i} X^-_{m,n_0,i} + Q^+_{n_0+1,i} X^+_{m,n_0+1,i}\right], \quad (72)$$

5. 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. While this isn't necessarily a problem for the inferred results, in this section we discuss an exact physical interpretation of the *celerite* kernel that will be 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 due 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

$$\left[\frac{\mathrm{d}^2}{\mathrm{d}t^2} + \frac{\omega_0}{Q}\,\frac{\mathrm{d}}{\mathrm{d}t} + \omega_0^2\right]\,y(t) = \epsilon(t) \tag{73}$$

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)

$$S(\omega) = \sqrt{\frac{2}{\pi}}\,\frac{S_0\,\omega_0^4}{(\omega^2-\omega_0^2)^2 + \omega_0^2\omega^2/Q^2} \tag{74}$$

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 (74) matches Equation (9) if

$$\begin{aligned}
a_j &= S_0\,\omega_0\,Q & (75)\\
b_j &= \frac{S_0\,\omega_0\,Q}{\sqrt{4\,Q^2-1}} & (76)\\
c_j &= \frac{\omega_0}{2\,Q} & (77)\\
d_j &= \frac{\omega_0}{2\,Q}\sqrt{4\,Q^2-1} \quad , & (78)
\end{aligned}$$

for $Q \geq \frac{1}{2}$. For $0 < Q \leq \frac{1}{2}$, Equation (74) can be captured by a pair of *celerite* terms with parameters

$$\begin{aligned}
a_{j\pm} &= \frac{1}{2}\,S_0\,\omega_0\,Q\,\left[1 \pm \frac{1}{\sqrt{1-4\,Q^2}}\right] & (79)\\
b_{j\pm} &= 0\\
c_{j\pm} &= \frac{\omega_0}{2\,Q}\,\left[1 \mp \sqrt{1-4\,Q^2}\right]\\
d_{j\pm} &= 0 \quad .
\end{aligned}$$

For these definitions, the kernel is

$$k_{\rm SHO}(\tau;\, S_0,\, Q,\, \omega_0) = S_0\,\omega_0\,Q\,e^{-\frac{\omega_0\,\tau}{2Q}} \begin{cases} \cosh\left(\eta\,\omega_0\,\tau\right) + \frac{1}{2\,\eta\,Q}\,\sinh\left(\eta\,\omega_0\,\tau\right), & 0 < Q < 1/2 \\ 2\,(1+\omega_0\,\tau), & Q = 1/2 \\ \cos\left(\eta\,\omega_0\,\tau\right) + \frac{1}{2\,\eta\,Q}\,\sin\left(\eta\,\omega_0\,\tau\right), & 1/2 < Q \end{cases} \tag{80}$$

where $\eta = |1 - (4\,Q^2)^{-1}|^{1/2}$. We note that, 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 (74) has several limits of physical interest:

- For $Q = 1/\sqrt{2}$, Equation (74) simplifies to

$$S(\omega) = \sqrt{\frac{2}{\pi}}\,\frac{S_0}{(\omega/\omega_0)^4 + 1} \quad . \tag{81}$$

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

$$k(\tau) = S_0\,\omega_0\,e^{-\frac{1}{\sqrt{2}}\,\omega_0\,\tau}\,\cos\left(\frac{\omega_0\,\tau}{\sqrt{2}} - \frac{\pi}{4}\right) \quad . \tag{82}$$

- Substituting $Q = 1/2$, Equation (74) becomes

$$S(\omega) = \sqrt{\frac{2}{\pi}}\,\frac{S_0}{\left[(\omega/\omega_0)^2 + 1\right]^2} \tag{83}$$

  with the corresponding covariance function (using Equation 8 and Equation 79)

$$\begin{aligned} k(\tau) &= \lim_{f\to 0}\,\frac{1}{2}\,S_0\,\omega_0\,\left[(1+1/f)\,e^{-\omega_0\,(1-f)\,\tau} + (1-1/f)\,e^{-\omega_0\,(1+f)\,\tau}\right] & (84)\\ &= S_0\,\omega_0\,e^{-\omega_0\,\tau}\,[1+\omega_0\,\tau] & (85)\end{aligned}$$

  or equivalently (using Equation 8 and Equation 75)

$$\begin{aligned} k(\tau) &= \lim_{f\to 0}\,S_0\,\omega_0\,e^{-\omega_0\,\tau}\,\left[\cos(f\,\tau) + \frac{\omega_0}{f}\,\sin(f\,\tau)\right] & (86)\\ &= S_0\,\omega_0\,e^{-\omega_0\,\tau}\,[1+\omega_0\,\tau] \quad . & (87)\end{aligned}$$

  This covariance function is also known as the Matérn-3/2 function (Rasmussen & Williams 2006). This suggests that the Matérn-3/2 covariance can be approximated using the *celerite* framework with a small value of f in Equation (86) but we caution that this might lead to numerical issues for the solver.

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

$$k(\tau) \approx S_0\,\omega_0\,Q\,\exp\left(-\frac{\omega_0\,\tau}{2\,Q}\right)\,\cos\left(\omega_0\,\tau\right) \quad . \tag{88}$$

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 $Q \leq 1/2$, 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 simple harmonic oscillator provides a physically motivated model that is flexible enough to describe a wide range of stellar variations and we return to give some specific examples in Section **??**. 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 previous section.

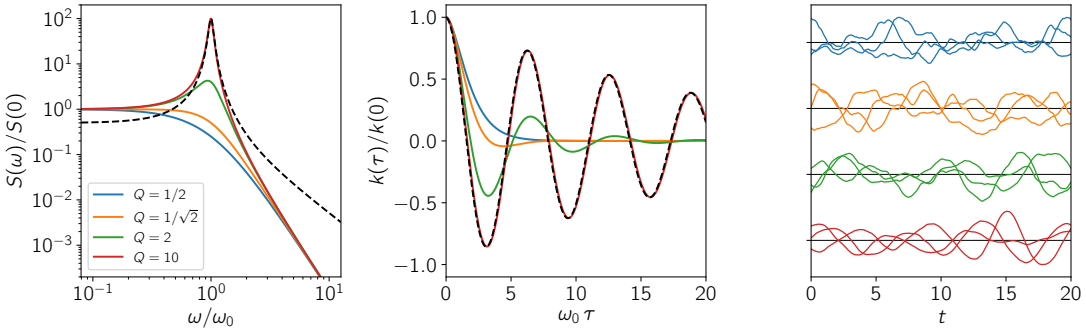

Figure 1. (left) The power spectrum of a stochastically-driven simple harmonic oscillator (Equation 74) plotted for several values of the quality factor Q. For comparison, the dashed line shows the Lorentzian function from Equation (11) with $c_j = \omega_0/(2\,Q) = 1/20$ and normalized so that $S(d_j)/S(0) = 100$. (middle) The corresponding autocorrelation functions with the same colors. (right) Three realizations from each model in the same colors.

6. EXAMPLES

To show 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 application of celerite for a range of use cases, 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 like, for example, 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 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

$$p(\boldsymbol{\theta}, \boldsymbol{\alpha} \,|\, \boldsymbol{y}, X) \propto p(\boldsymbol{y} \,|\, X, \boldsymbol{\theta}, \boldsymbol{\alpha})\, p(\boldsymbol{\theta}, \boldsymbol{\alpha}) \tag{89}$$

where $p(\boldsymbol{y} \,|\, X, \boldsymbol{\theta}, \boldsymbol{\alpha})$ 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 separable uniform priors with plausibly broad support.

6.1. *Example 1: Recovery of a celerite process*

In this first example, we simulate a dataset using a known *celerite* process and fit it with celerite to show that valid inferences can be made in this idealized case. We simulated a small ($N = 200$) dataset using a GP model with a SHO kernel (Equation 80) with parameters $S_0 = 1\,\mathrm{ppm}^2$, $\omega_0 = e^2\,\mathrm{day}^{-1}$, and $Q = e^2$, where the units are arbitrarily chosen to simplify the discussion. We add further white noise with the amplitude $\sigma_n = 2.5\,\mathrm{ppm}$ to each data point. The simulated data are plotted in the top left panel of Figure 2.

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 (80) 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 non-linear 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 2 shows the conditional mean and standard deviation of the MAP model over-plotted 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 that the residuals appear qualitatively uncorrelated in this figure.

We then sample the joint posterior probability (Equation 89) using emcee (Goodman & Weare 2010; Foreman-Mackey et al. 2013). We initialize 32 walkers in a Gaussian

Table 1. The parameters and priors for Example 1.

parameter	prior
$\ln(S_0)$	$\mathcal{U}(-10, 10)$
$\ln(Q)$	$\mathcal{U}(-10, 10)$
$\ln(\omega_0)$	$\mathcal{U}(-10, 10)$

ball with width 10^{-4} in each parameter around the MAP parameter vector, run 500 steps of burn-in, and 2000 steps of MCMC. To assess convergence, we estimate the mean integrated autocorrelation time of the chain 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-hand panel of Figure 2. In this figure, the true PSD is plotted as a dashed black line and the numerical estimate of the posterior inference 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.

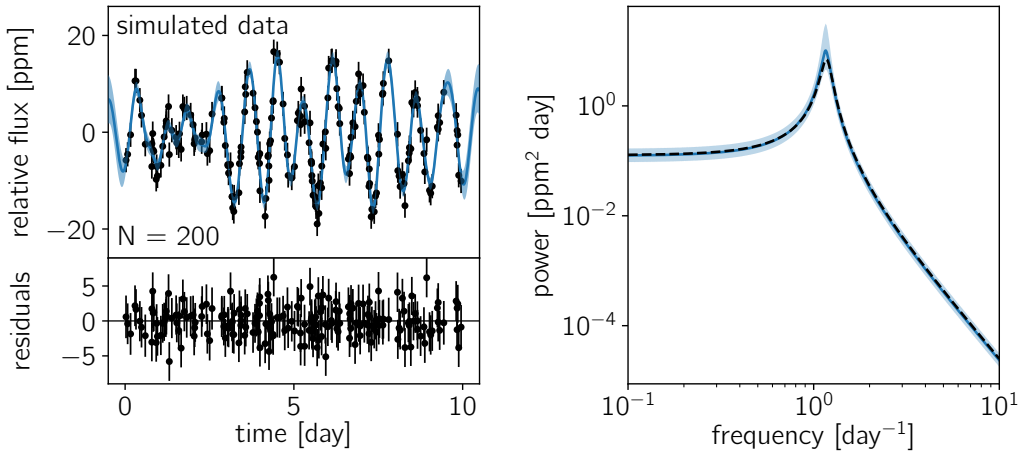

Figure 2. (top left) A simulated dataset (black error bars), and maximum likelihood model (blue contours). (bottom left) The residuals between the mean predictive model and the data shown in the top left figure. (right) The inferred PSD – the blue contours encompass 68% of the posterior mass – compared to the true PSD (dashed black line).

6.2. *Example 2: Inferences with the "wrong" model*

In this example, we simulate a dataset using a known GP model with a kernel outside of the support of a *celerite* process. This means 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 quasiperiodic GP with the kernel

$$k_{\rm true}(\tau) = A\,\exp\left(-\frac{\tau^2}{2\,\lambda^2}\right)\,\cos\left(\frac{2\,\pi\,\tau}{P_{\rm true}}\right) \tag{90}$$

where $P_{\rm true}$ is the fundamental period of the process. This autocorrelation structure corresponds to the power spectrum (Wilson & Adams 2013)

$$S_{\rm true}(\omega) = \frac{\lambda\,A}{2}\left[\exp\left(-\frac{\lambda^2}{2}\left(\omega - \frac{2\,\pi}{P_{\rm true}}\right)^2\right) + \exp\left(-\frac{\lambda^2}{2}\left(\omega + \frac{2\,\pi}{P_{\rm true}}\right)^2\right)\right] \tag{91}$$

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 $P_{\rm true}$ even with this effective model.

We generate a realization of a GP model with the kernel given in Equation (90) with $N = 100$, $A = 1\,{\rm ppm}^2$, $\lambda = 5\,{\rm day}$, and $P_{\rm true} = 1\,{\rm day}$. We also add white noise with amplitude $\sigma = 0.5\,{\rm ppm}$ to each data point. The top left panel of Figure 3 shows this simulated dataset.

We then fit this simulated data using the product of two SHO terms (Equation 80) where one of the terms has $S_0 = 1$ and $Q = 1/\sqrt{2}$ fixed. The kernel for this model is

$$k(\tau) = k_{\rm SHO}(\tau;\,S_0,\,Q,\,\omega_1)\,k_{\rm SHO}(\tau;\,S_0 = 1,\,Q = 1/\sqrt{2},\,\omega_2) \tag{92}$$

where $k_{\rm SHO}$ is defined in Equation (80) 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.

As in the previous example, we set the mean function to zero and can, therefore, omit the parameters $\boldsymbol{\theta}$. Table 2 lists the proper log-uniform priors that we choose for each parameter in $\boldsymbol{\alpha} = (S_0,\,Q,\,\omega_1,\,\omega_2)$. 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 3 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 4-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 effective 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

Table 2. The parameters and priors for Example 2.

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

kernel given by Equation (90) 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 3. 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 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.

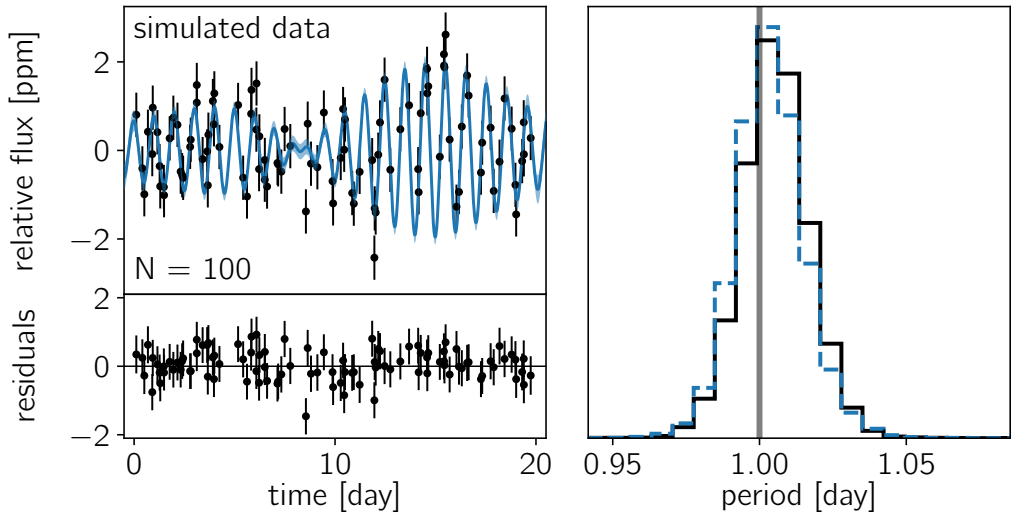

Figure 3. (top left) A simulated dataset (black error bars), and maximum likelihood model (blue contours). (bottom left) The residuals between the mean predictive model and the data shown in the top left figure. (right) The inferred period of the process. The true period is indicated by the vertical orange 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.

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 quasiperiodic 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 quasiperiodic covariance function can be used to make probabilistic measurements even with sparsely sampled data (R. Angus, *et al.* submitted). The covariance function used for this type of analysis has the form

$$k(\tau) = A \exp\left(-\frac{\tau^2}{2\,\ell^2} - \Gamma \sin^2\left(\frac{\pi\,\tau}{P_\mathrm{rot}}\right)\right) \tag{93}$$

where P_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 quasiperiodic photometric time series in general (Wang et al. 2012). The key difference between Equation (93) and other quasiperiodic kernels is that it is positive for all values of τ. We construct a simple *celerite* covariance function with similar properties as follows

$$k(\tau) = \frac{B}{2+C}\,e^{-\tau/L}\left[\cos\left(\frac{2\,\pi\,\tau}{P_\mathrm{rot}}\right) + (1+C)\right] \tag{94}$$

for $B > 0$, $C > 0$, and $L > 0$. The covariance function in Equation (94) cannot exactly reproduce Equation (93) but, since Equation (93) is only an effective model, Equation (94) 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 datasets (for example 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 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 (94) 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).

We select about 180 days of contiguous observations of KIC 1430163 from Kepler. This dataset has 6950 measurements and, using a tuned linear algebra implementation[3], a single evaluation of the likelihood requires over 8 seconds on a modern Intel CPU. This calculation, using celerite with the model in Equation (94), only takes ~ 1.5 ms – a speed-up of more than three orders of magnitude per model evaluation.

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

Table 3. The parameters and priors for Example 3.

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

We set the mean function $\mu_{\boldsymbol{\theta}}$ to zero and the remaining parameters $\boldsymbol{\alpha}$ and their priors are listed in Table 3. As with the earlier examples, we start by estimating the MAP parameters 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 4 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 $\ln P_{\mathrm{rot}}$ and estimate that we have 2900 independent samples. These samples give a posterior constraint on the period of $P_{\mathrm{rot}} = 3.80 \pm 0.15$ and the marginalized posterior distribution for P is shown in the right panel of Figure 4. 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 forthcoming work (R. Angus, *et al.*, submitted) demonstrates that GP inferences are, at a population level, more reliable than other methods.

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; Huber et al. 2011; Chaplin et al. 2011, 2013; Stello et al. 2013) and asteroseismology is a key science driver for many of the upcoming large scale photometric surveys (Campante et al. 2016; Rauer et al. 2014; Gould et al. 2015). Most asteroseismic analyses have been limited to high signal-to-noise oscillations because the standard methods use statistics of the empirical periodogram and cannot formally propagate the measurement uncertainties to the constraints on physical parameters, instead relying on bootstrapped uncertainty estimators (Huber et al. 2009). More sophisticated methods that compute the likelihood function in the time domain scale poorly to large survey datasets (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 datasets. In practice, we model the star as a mixture of stochastically-driven simple harmonic oscillators where the amplitudes and frequencies of the oscillations are computed using a physical model, and

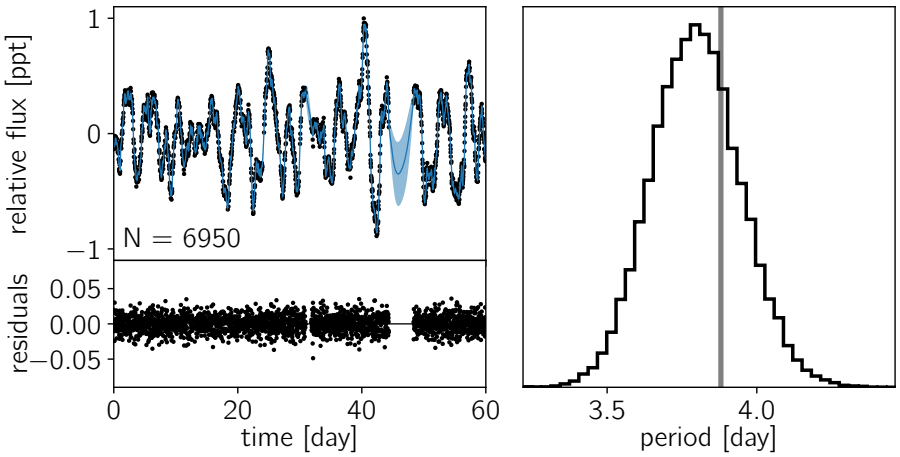

Figure 4. Inferred constraints on a quasiperiodic GP model using the covariance function in Equation (94) and two quarters of Kepler data. (top left) The Kepler data (black points) and the maximum likelihood model prediction (blue curve) for a 60 day subset of the data used. The solid blue line shows the predictive mean and the blue contours show the predictive standard deviation. (bottom left) The residuals between the mean predictive model and the data shown in the top left figure. (right) The posterior constraint on the rotation period of KIC 1430163 using the dataset and model from Figure 4. The period is the parameter $P_{\rm rot}$ in Equation (94) and this figure shows the posterior distribution marginalized over all other nuisance parameters in Equation (94). This is consistent with the published rotation period made using the full Kepler baseline shown as a vertical gray line (Mathur et al. 2014).

evaluate the probability of the observed time series using a GP where the PSD is a sum of terms given by Equation (74). This gives us a method for computing the likelihood function for the parameters of the physical model (for example, $\nu_{\max}$ and $\Delta\nu$, or other more fundamental parameters) *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 datasets 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 8 components with amplitudes and frequencies specified by $\nu_{\max}$, $\Delta\nu$, and some nuisance parameters. The first term is used to capture the granulation "background" (Kallinger et al. 2014) using Equation (81) with two free parameters S_g and ω_g. The remaining 7 terms are given by Equation (74) where Q is a nuisance parameter shared between terms and the frequencies are given by

$$\omega_{0,\,j} = 2\,\pi\,(\nu_{\max} + j\,\Delta\nu + \epsilon) \tag{95}$$

and the amplitudes are given by

$$S_{0,\,j} = \frac{A}{Q^2}\,\exp\left(-\frac{[j\,\Delta\nu + \epsilon]^2}{2\,W^2}\right) \tag{96}$$

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 $\nu_{\max}$ and ϵ. To build a more realistic model, this prescription could be extended to include more angular modes, or $\nu_{\max}$ 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, observed by the Kepler Mission. The goal of this example is to show that, even for a low signal-to-noise dataset 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 dataset. Looking forward to TESS (Ricker et al. 2014; Campante et al. 2016), we measure $\nu_{\max}$ and $\Delta\nu$ using only one month of Kepler data and compare our results to the results inferred from the full 4 year baseline of the Kepler Mission. For KIC 11615890, the published asteroseismic parameters measured using several years of Kepler observations are (Pinsonneault et al. 2014)

$$\nu_{\max} = 171.94 \pm 3.62\,\mu\text{Hz} \quad \text{and} \quad \Delta\nu = 13.28 \pm 0.29\,\mu\text{Hz} \quad . \tag{97}$$

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 (Stumpe et al. 2012; Smith et al. 2012) Kepler data. Unlike the previous examples, the posterior distribution is sharply multimodal and naïvely maximizing the posterior using L-BFGS-B is not practical. Instead, we start by estimating the initial values for $\nu_{\max}$ 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 set of L-BFGS-B optimizations with values of $\nu_{\max}$ selected in logarithmic grid centered on our initial estimate of $\nu_{\max}$ 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 7 shows a 10 day subset of the dataset 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 7. 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 15000 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 6 shows the marginalized density for $\nu_{\max}$ 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. All asteroseismic analyses are known to have substantial systematic and method-dependent uncertainties (Verner et al. 2011) so further experiments would be needed to fully assess the reliability of this method.

This model requires about 10 CPU minutes to run the MCMC to convergence. 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[4] 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.

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

Table 4. The parameters and priors for Example 4.

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

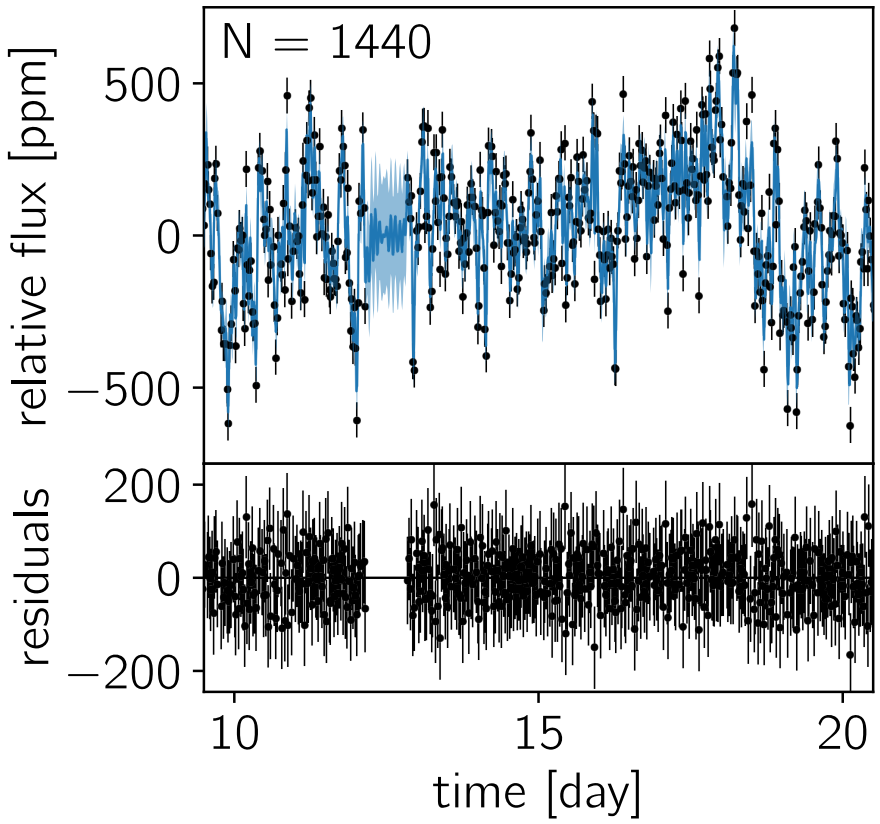

Figure 5. (top) The Kepler data (black points) and the maximum likelihood model prediction (blue curve) for a 10 day subset of the month-long dataset that was used for the fit. The solid blue line shows the predictive mean and the blue contours show the predictive standard deviation. (bottom) The residuals between the mean predictive model and the data shown in the top figure.

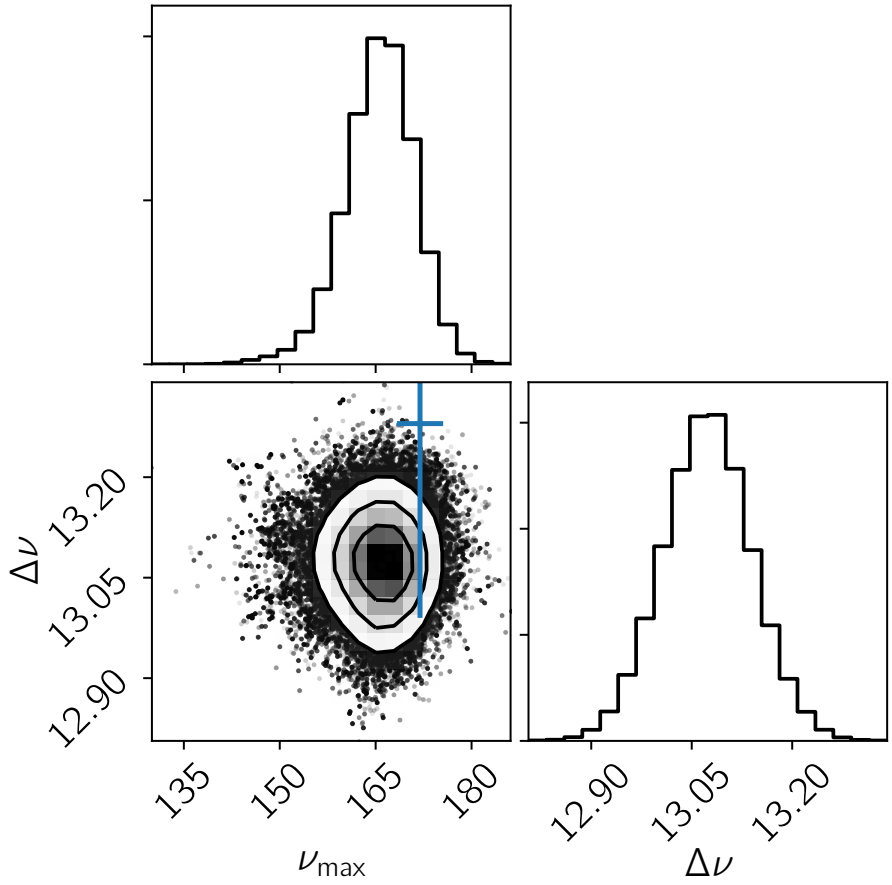

Figure 6. The probabilistic constraints on ν_{max} and $\Delta\nu$ from the inference shown in Figure 7 compared to the published value (error bar) based on several years of Kepler observations (Pinsonneault et al. 2014). The two-dimensional contours show the 0.5-, 1-, 1.5, and 2-sigma credible regions in the marginalized planes and the histograms along the diagonal show the marginalized posterior for each parameter.

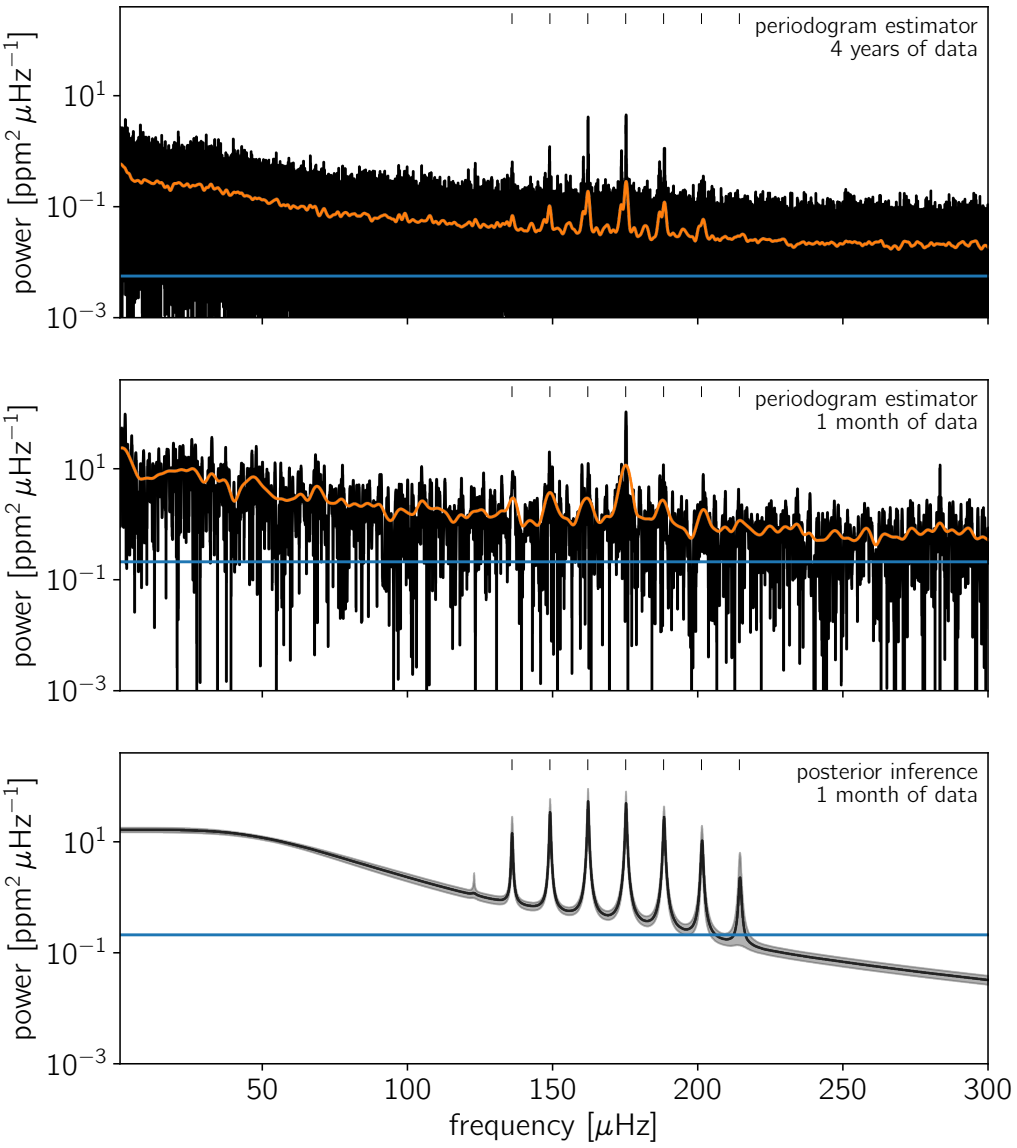

Figure 7. A comparison between the Lomb-Scargle estimator of the PSD and the posterior inference of the PSD as a mixture of stochastically-driven simple harmonic oscillators. (top) The periodogram of the Kepler light curve for KIC 11615890 computed on the full four year baseline of the mission. The orange line shows a smoothed periodogram and the blue line indicates the level of the measurement uncertainties. (middle) The same periodogram computed using about a month of data. (bottom) The power spectrum inferred using the mixture of SHOs model described in the text and only one month of Kepler data. The black line shows the median of posterior PSD and the gray contours show the 68% credible region.

6.5. *Example 5: Exoplanet transit fitting*

This example is different from all the previous examples – both simulated and real – because, in this case, do not set the deterministic mean function $\mu_{\boldsymbol{\theta}}$ to zero. Instead, we make inferences about $\mu_{\boldsymbol{\theta}}$ 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 (for example 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 $\boldsymbol{\theta}$ and, in this example, the mean function $\mu_{\boldsymbol{\theta}}(t)$ is a limb-darkened transit light curve (Mandel & Agol 2002; Foreman-Mackey & Morton 2016) and the parameters $\boldsymbol{\theta}$ are the orbital period $P_{\rm orb}$, 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 (94) and fit for the parameters of the exoplanet $\boldsymbol{\theta}$ and the stellar variability $\boldsymbol{\alpha}$ simultaneously. The full set of parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\theta}$ are listed in Table 5 along with their priors and the true values for $\boldsymbol{\theta}$.

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 $\boldsymbol{\alpha}$ and $\boldsymbol{\theta}$ simultaneously. The top panel of Figure 8 shows the data including the simulated transit as black points with the MAP model prediction over-plotted in blue. The bottom panel of Figure 8 shows the "de-trended" light curve where the MAP model has been subtracted and the MAP mean model $\mu_{\boldsymbol{\theta}}$ has been added back in. For comparison, the transit model is over-plotted in the bottom panel of Figure 8 and we see no obvious correlations in the residuals.

Sampling 32 walkers from an isotropic Gaussian centered on the MAP parameters, we run 10000 steps of burn-in and 30000 steps of production MCMC using emcee. We estimate the integrated autocorrelation time for the $\ln(P_{\rm orb})$ chain and find 1490 effective samples across the full chain. Figure 9 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 $\boldsymbol{\theta}$ are consistent with the true values. This promising result suggests that celerite can be used as an

Table 5. The parameters, priors, and (if known) the true values used for the simulation in Example 5.

	parameter	prior	true value
kernel: $\boldsymbol{\alpha}$	$\ln(B/\mathrm{ppt}^2)$	$\mathcal{U}(-10.0,\, 0.0)$	—
	$\ln(L/\mathrm{day})$	$\mathcal{U}(1.5,\, 5.0)$	—
	$\ln(P_\mathrm{rot}/\mathrm{day})$	$\mathcal{U}(-3.0,\, 5.0)$	—
	$\ln(C)$	$\mathcal{U}(-5.0,\, 5.0)$	—
	$\ln(\sigma/\mathrm{ppt})$	$\mathcal{U}(-5.0,\, 0.0)$	—
mean: $\boldsymbol{\theta}$	f_0/ppt	$\mathcal{U}(-0.5,\, 0.5)$	0
	$\ln(P_\mathrm{orb}/\mathrm{day})$	$\mathcal{U}(\ln(7.9),\, \ln(8.1))$	$\ln(8)$
	$\ln(R_p/R_\star)$	$\mathcal{U}(\ln(0.005),\, \ln(0.1))$	$\ln(0.015)$
	$\ln(T/\mathrm{day})$	$\mathcal{U}(\ln(0.4),\, \ln(0.6))$	$\ln(0.5)$
	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

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 this section, we demonstrated five potential use cases for celerite. These examples span a range of data sizes and model complexities, and the last three are all based on active areas of research with immediate real-world applicability. The sizes of these datasets are modest on the scale of some existing and forthcoming survey datasets 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 Example 3, for example, we inferred the rotation period of a star observed by Kepler with a cost of about 4 CPU-minutes. By contrast, the same inference using a general but optimized Cholesky factorization routine would require nearly 400 CPU-hours. The upcoming TESS Mission will observe *DFM*: XXX stars.

7. COMPARISONS TO OTHER METHODS

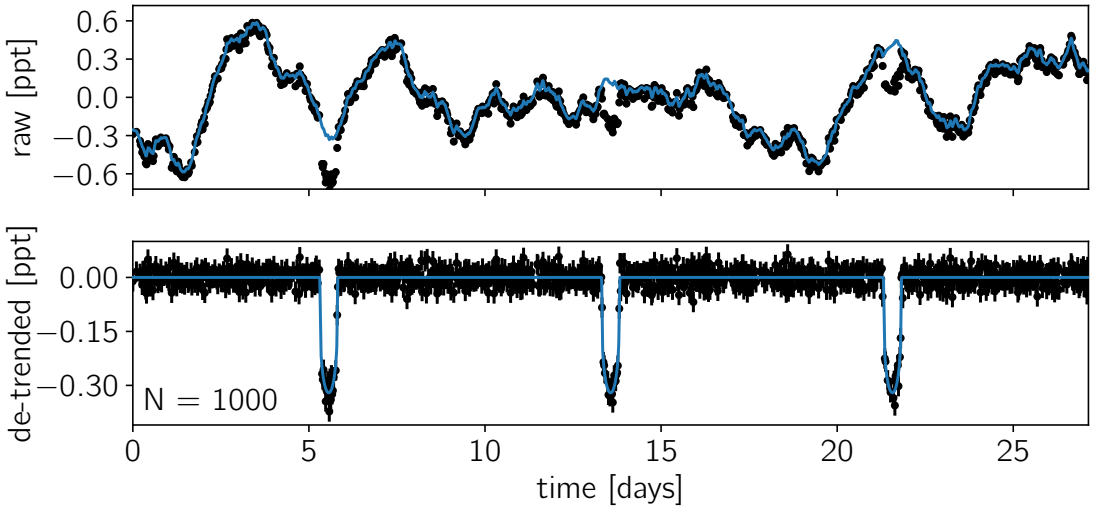

Figure 8. *(top)* A month-long segment of Kepler light curve for KIC 1430163 with a synthetic transit model injected (black points) and the maximum likelihood model for the stellar variability (blue line). *(bottom)* The maximum likelihood "de-trending" of the data in the top panel. In this panel, the maximum likelihood model for the stellar variability has been subtracted to leave only the transits. The de-trended light curve is shown by black error bars and the maximum likelihood transit model is shown as a blue line.

Table 6. The 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	80000	1737
2	100	2	0.67	0.36	4	80000	1134
3	6950	2	8119.11	1.47	4	176000	2900
4	1440	8	828.93	2.74	9	640000	1443
5	1000	2	88.30	0.96	13	1280000	1490

[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.

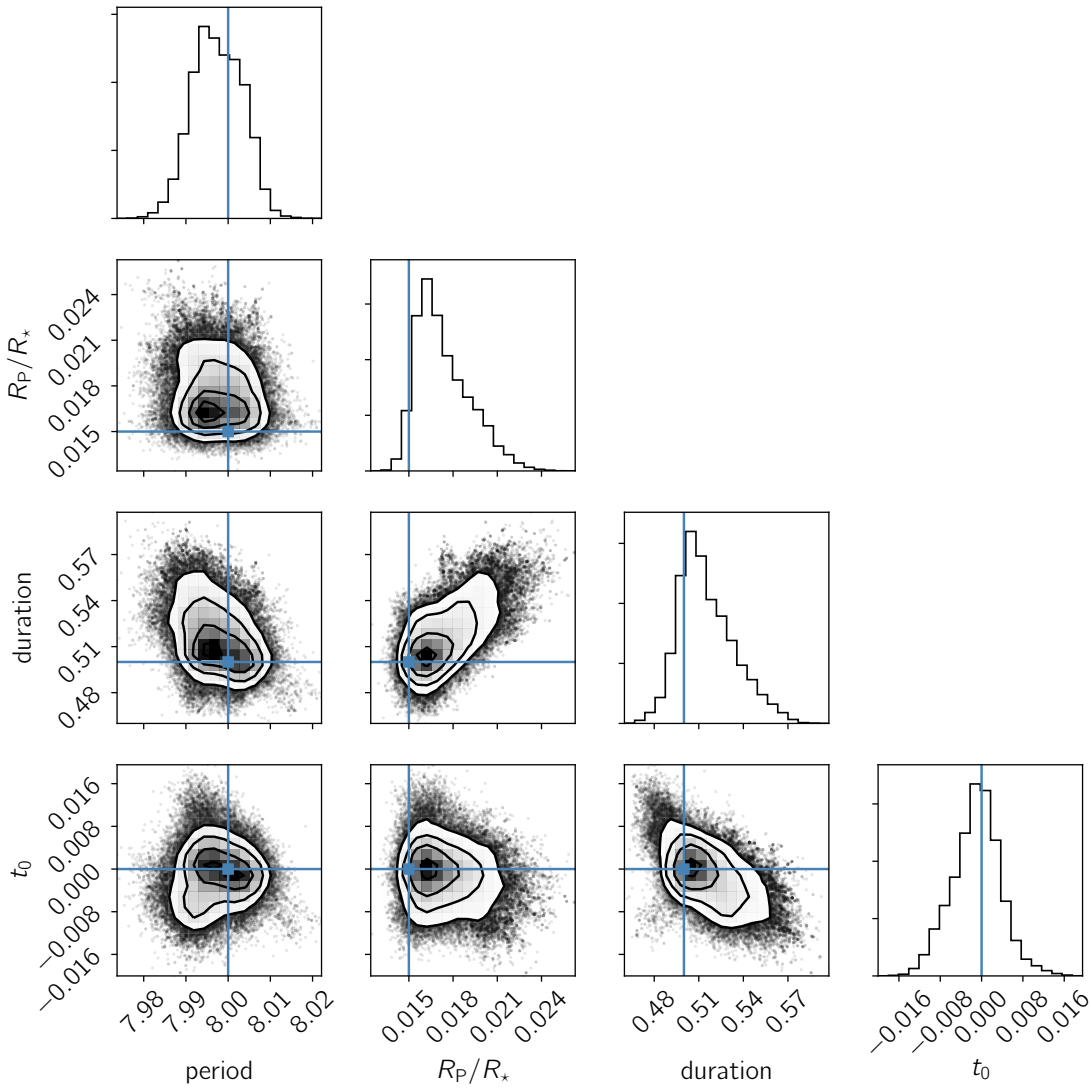

Figure 9. The marginalized posterior constraints on the physical parameters of the planet transit in the light curve shown in the top panel of Figure 8. The two-dimensional contours show the 0.5-, 1-, 1.5, and 2-sigma credible regions in the marginalized planes and the histograms along the diagonal show 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.

There are other methods of scaling GP models to large datasets 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 it can only be used in one-dimension.

7.1. *Kalman filtering & CARMA models*

The continuous autoregressive moving average (CARMA) models introduced into the astrophysics literature by Kelly et al. (2014) share many features with celerite. CARMA models are derived by solving a stochastic differential equation and, like celerite, the likelihood function for a CARMA model conditioned on a one-dimensional dataset can be solved in $\mathcal{O}(N)$. The kernel function for a CARMA(J, K) model is (Kelly et al. 2014)

$$k_{\rm CARMA}(\tau) = \sum_{j=1}^{J} A_j\, \exp\left(r_j\,\tau\right) \tag{98}$$

where

$$A_j = \sigma^2\, \frac{\left[\sum_{k=0}^{K} \beta_k\, (r_j)^k\right]\, \left[\sum_{k=0}^{K} \beta_k\, (-r_j)^k\right]}{-2\,\mathrm{Re}(r_j)\, \prod_{k=1,\,k\neq j}^{J} (r_k - r_j)(r_k{}^* + r_j)} \tag{99}$$

and σ, $\{r_j\}_{j=1}^J$ and $\{\beta_k\}_{k=1}^K$ are parameters of the model. Comparing Equation (98) 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. 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 (99) numerically for a given set of $\{A_j\}_{j=1}^J$. While the computational scaling of CARMA models is also $\mathcal{O}(N\,J^2)$, the Kalman filtering based method for solving CARMA models (Kelly et al. 2014) is, in practice, somewhat faster and the memory requirements are smaller. One benefit of the *celerite* model is that it is parameterized in terms of the amplitudes directly. This means that the physical interpretation of these parameters is, in some cases, simpler. On the other hand, CARMA models are parameterized by modeling the power spectrum of the stochastic driving force and this might be the quantity of interest in other situations. Another benefit of the *celerite* framework is that, unlike filtering-based CARMA solvers, it directly solves matrix equations of the form ${K_{\boldsymbol{\alpha}}}^{-1}\,b$ in $\mathcal{O}(N)$. This is a crucial feature when GP modeling is combined with linear models (for example Luger et al. 2017).

7.2. *Other scalable methods*

Another popular method uses the fact that, in the limit of evenly-spaced data and homoscedastic uncertainties, the covariance matrix is "Toeplitz" (for example

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 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 datasets.

Carter & Winn (2009) improved the scaling of Toeplitz methods by introducing a wavelet-based method for computing 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, Ambikasaran et al. 2016) solver. 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 multi-dimensional 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 for small models. This means that the HODLR solver can *approximately* evaluate *celerite* models more efficiently than celerite for large models $J \gg 10$ and small datasets where N is smaller than a few thousand.

The structured kernel interpolation (SKI/KISS-GP Wilson & Nicisch 2015) framework is another approximate method that can be used to scale GPs to large datasets. 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 dataset, 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, for example, 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, but flexible, class of one-dimensional kernel functions. Furthermore, since *celerite* models can be interpreted as a mixture of stochastically-driven, damped simple harmonic oscillators, they are a physically motivated choice of

covariance function in many applications.

8. SUMMARY

Gaussian Process models have been fruitfully applied to many problems in astronomical data analysis, but the fact that the computational cost scales as the cube of the number of data points has limited their use to relatively small datasets. With the linear scaling of celerite we envision the application of Gaussian processes may be expanded to much larger datasets. 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. A drawback of this method is its quadratic scaling with the number of terms J, but, in many cases, small values of J are sufficient.

Our background is in studying transiting exoplanets, a field which has only recently begun to adopt full covariance matrices in analyzing the noise in stellar light curves when detecting or characterizing transiting planets (for example, Carter & Winn 2009; Gibson et al. 2012; Barclay et al. 2015; Evans 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 datasets or restrictive kernel choices. celerite weakens these requirements by providing a scalable method for computing the likelihood and a physical motivation for the choice of kernel. celerite can be used to model stellar oscillations using the relation to the mixture of stochastically-driven, damped simple harmonic oscillators. 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 expect that celerite will be important for transit detection (Pope et al. 2016; Foreman-Mackey 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), phase functions (Knutson et al. 2007; Zucker et al. 2007), and more.

Beyond these applications to model stellar variability, the method is generally applicable to other one-dimensional GP models. Accreting black holes show time series which may be modeled using a GP (Kelly et al. 2014); indeed, this was the motivation for the original technique developed by Rybicki & Press (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. In principle, a large number of *celerite* terms could be used to perform non-parametric probabilistic inference of the power spectrum despite unevenly-spaced data with heteroscedastic noise (for example, 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 (CARMA models, Kelly et al. 2014). In general, we encourage the use of physically motivated models for parameter estimation instead of qualitative modeling of the power spectrum itself.

There are many data analysis problems where celerite will not be applicable. In particular, the restriction to one-dimensional problems is significant. There are many examples of multidimensional GP modeling 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 accelerate any of these analyses. It is plausible that an extension could 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.

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`[5] and Zenodo (Foreman-Mackey et al. 2017), and it is made available under the MIT license.

It is a pleasure to thank Megan Bedell, Ian Czekala, Will Farr, Sam Grunblatt, David W. Hogg, Dan Huber, Meredith Rawls, Dennis Stello, Jake VanderPlas, and Andrew Gordon Wilson for helpful discussions informing the ideas and code presented here. We would also like to thank the anonymous referee for thorough and constructive feedback that greatly improved the paper.

This work was performed in part under contract with the Jet Propulsion Laboratory (JPL) funded by NASA through the Sagan Fellowship Program executed by the NASA Exoplanet Science Institute. EA acknowledges support from NASA grants NNX13AF20G, NNX13A124G, NNX13AF62G, from National Science Foundation (NSF) grant AST-1615315, and from NASA Astrobiology Institute's Virtual Planetary Laboratory, supported by NASA under cooperative agreement NNH05ZDA001C.

This research made use of the NASA Astrophysics Data System and the NASA Exoplanet Archive. The Exoplanet Archive is operated by the California Institute of Technology, under contract with NASA under the Exoplanet Exploration Program.

This paper includes data collected by the Kepler Mission. Funding for the Kepler Mission is provided by the NASA Science Mission directorate. We are grateful

[5] This version of the paper was generated with git commit `7f4ea50` (2017-06-05).

to the entire Kepler team, past and present. These data were obtained from the Mikulski Archive for Space Telescopes (MAST). STScI is operated by the Association of Universities for Research in Astronomy, Inc., under NASA contract NAS5-26555. Support for MAST is provided by the NASA Office of Space Science via grant NNX13AC07G and by other grants and contracts.

This research made use of Astropy, a community-developed core Python package for Astronomy (Astropy Collaboration et al. 2013).

Facility: Kepler

Software: AstroPy (Astropy Collaboration et al. 2013), corner.py (Foreman-Mackey 2016), Eigen (Guennebaud et al. 2010), emcee (Foreman-Mackey et al. 2013), george (Ambikasaran et al. 2016), LAPACK (Anderson et al. 1999), matplotlib (Hunter et al. 2007), numpy (Van Der Walt et al. 2011), transit (Foreman-Mackey & Morton 2016), scipy (Jones et al. 2001).

APPENDIX

A. AN ALTERNATIVE SOLVER USING THE EXTENDED MATRIX FORMALISM

Rybicki & Press (1995) demonstrated that the inverse of a matrix K, with elements given by Equation (5), could be computed efficiently by taking advantage of the structure of this covariance function. Ambikasaran (2015) generalized this computation to apply to the full mixture of J terms in Equation (7) and derived an equally efficient method for computing the determinant of K.

A.1. *An example*

To provide some insight for this method, we follow Ambikasaran (2015) and start by working through a simple example. In this case, we assume that we have three data points $\{y_1, y_2, y_3\}$ observed at times $\{t_1, t_2, t_3\}$ with measurement variances $\{{\sigma_1}^2, {\sigma_2}^2, {\sigma_3}^2\}$ and we would like to compute the likelihood of these data under a GP model with the covariance function

$$k(\tau_{nm}) = \sigma_n^2\,\delta_{nm} + a\,\exp(-c\,\tau_{nm}) \quad . \tag{A1}$$

This is the result of setting $J = 1$, $b = 0$, and $d = 0$ in Equation (7) and it is the model studied by Rybicki & Press (1995). To demonstrate that the likelihood of this model can be computed in $\mathcal{O}(N)$, we write out the full system of equations that must be solved to apply the inverse of K and compute the first term of Equation (3). In matrix notation, this is

$$K\,\boldsymbol{z} = \boldsymbol{y}, \tag{A2}$$

$$\begin{pmatrix} a + {\sigma_1}^2 & a\,e^{-c\,\tau_{2,1}} & a\,e^{-c\,\tau_{3,1}} \\ a\,e^{-c\,\tau_{2,1}} & a + {\sigma_2}^2 & a\,e^{-c\,\tau_{3,2}} \\ a\,e^{-c\,\tau_{3,1}} & a\,e^{-c\,\tau_{3,2}} & a + {\sigma_3}^2 \end{pmatrix} \begin{pmatrix} z_1 \\ z_2 \\ z_3 \end{pmatrix} = \begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} \tag{A3}$$

where our goal is to solve for the unknown vector $\boldsymbol{z}$ for a given matrix K and vector $\boldsymbol{y}$. In Equation (A2), we have assumed that the mean function is zero but a non-zero mean could be included by replacing $\boldsymbol{y}$ by $\boldsymbol{r_\theta}$ as defined in Section 2. Now, if we introduce the variables

$$g_n = e^{-c\,\tau_{n+1,n}}\,g_{n-1} + e^{-c\,\tau_{n+1,n}}\,z_n \tag{A4}$$

where $g_0 = 0$, and

$$u_n = e^{-c\,\tau_{n+2,n+1}}\,u_{n+1} + a\,z_{n+1} \tag{A5}$$

where $u_N = 0$, the system of equations can be rewritten as

$$\begin{aligned}
(a + {\sigma_1}^2)\, z_1 + e^{-c\,\tau_{2,1}}\, u_1 &= y_1 && \text{(A6)}\\
a\, g_1 + (a + {\sigma_2}^2)\, z_2 + e^{-c\,\tau_{3,2}}\, u_2 &= y_2 && \text{(A7)}\\
a\, g_2 + (a + {\sigma_3}^2)\, z_3 &= y_3 \quad . && \text{(A8)}
\end{aligned}$$

Rewriting the system defined by Equation (A5) through Equation (A8) as a matrix equation shows the benefit that this seemingly trivial reformulation provides:

$$\begin{pmatrix}
a + {\sigma_1}^2 & e^{-c\,\tau_{2,1}} & 0 & 0 & 0 & 0 & 0 \\
e^{-c\,\tau_{2,1}} & 0 & -1 & 0 & 0 & 0 & 0 \\
0 & -1 & 0 & a & e^{-c\,\tau_{3,2}} & 0 & 0 \\
0 & 0 & a & a + {\sigma_2}^2 & e^{-c\,\tau_{3,2}} & 0 & 0 \\
0 & 0 & e^{-c\,\tau_{3,2}} & e^{-c\,\tau_{3,2}} & 0 & -1 & 0 \\
0 & 0 & 0 & 0 & -1 & 0 & a \\
0 & 0 & 0 & 0 & 0 & a & a + {\sigma_3}^2
\end{pmatrix}
\begin{pmatrix} z_1 \\ u_1 \\ g_1 \\ z_2 \\ u_2 \\ g_2 \\ z_3 \end{pmatrix}
=
\begin{pmatrix} y_1 \\ 0 \\ 0 \\ y_2 \\ 0 \\ 0 \\ y_3 \end{pmatrix}$$

Following Ambikasaran (2015) we call this the "extended" system and rewrite Equation (A2) as

$$K_{\text{ext}}\, \boldsymbol{z}_{\text{ext}} = \boldsymbol{y}_{\text{ext}} \quad . \tag{A9}$$

Even though K_{ext} is a larger matrix than the K we started with, it is now sparse with banded structure that can be exploited to solve the system efficiently. In particular, sparse solvers are available that can perform a LU-decomposition of matrices like this in $\mathcal{O}(N)$ operations – instead of the $\mathcal{O}(N^3)$ that would be required in general – and we can use these algorithms to solve our system exactly because the target vector $\boldsymbol{z}$ is a subset of the elements of $\boldsymbol{z}_{\text{ext}}$.

In the following section we discuss this method more generally, but it's worth noting a few important facts that can already be seen in this example. First, the fundamental reason why this matrix K can be solved efficiently is the following property of exponentials

$$e^{-c\,(t_3 - t_2)}\, e^{-c\,(t_2 - t_1)} = e^{-c\,(t_3 - t_2 + t_2 - t_1)} = e^{-c\,(t_3 - t_1)} \tag{A10}$$

and it is important to note that this property does not extend to other common covariance functions like the "exponential-squared" function

$$k(\tau) \propto e^{-c\,\tau^2} \quad . \tag{A11}$$

Second, our derivation of the extended matrix requires that the data points be monotonically sorted. This is the motivation for our previous statement that this method can only be applied in one dimension since neither of these properties will be satisfied by multidimensional inputs.

Ambikasaran (2015) demonstrated two key facts that allow us to use this extended matrix formalism in practice. First, even if the covariance function is a mixture of exponentials, the extended matrix will still be banded with a bandwidth that scales linearly with the number of components, J. Second, Ambikasaran (2015) proved that the absolute value of the determinant of $K_{\rm ext}$ is equal to the absolute value of the determinant of K. This means we can use this extended matrix formalism to compute the marginalized likelihood in $\mathcal{O}(N)$ operations.

A.2. *The algorithm*

In this section, we generalize the method from the previous section to the covariance function given by Equation (8). This derivation follows Ambikasaran (2015) but it includes explicit treatment of complex parameters, and their complex conjugates.

In the case of the full *celerite* covariance function (Equation 8), we introduce the following auxiliary variables in analogy to the u_n and g_n that we introduced in the previous section

$$\begin{aligned}
\phi_{n,j} &= e^{-c_j\,\tau_{n+1,n}}\,\cos\left(d_j\,\tau_{n+1,n}\right) && \text{(A12)}\\
\psi_{n,j} &= -e^{-c_j\,\tau_{n+1,n}}\,\sin\left(d_j\,\tau_{n+1,n}\right) && \text{(A13)}\\
g_{n,j} &= \phi_{n,j}\,g_{n-1,j} + \phi_{n,j}\,z_n + \psi_{n,j}\,h_{n-1,j} && \text{(A14)}\\
h_{n,j} &= \phi_{n,j}\,h_{n-1,j} - \psi_{n,j}\,z_n - \psi_{n,j}\,g_{n-1,j} && \text{(A15)}\\
u_{n,j} &= \phi_{n+1,j}\,u_{n+1,j} + a_j\,z_{n+1} + \psi_{n+1,j}\,v_{n+1,j} && \text{(A16)}\\
v_{n,j} &= \phi_{n+1,j}\,v_{n+1,j} - b_j\,z_{n+1} - \psi_{n+1,j}\,u_{n+1,j} && \text{(A17)}
\end{aligned}$$

with the boundary conditions

$$g_{0,j} = 0 \quad , \quad h_{0,j} = 0 \quad , \quad u_{N,j} = 0 \quad , \quad \text{and} \quad v_{N,j} = 0 \qquad \text{(A18)}$$

for all j. Using these variables and some algebra, we find that the following expression

$$\sum_{j=1}^{J}\left[a_j\,g_{n,j} + b_j\,h_{n,j}\right] + \left[{\sigma_n}^2 + \sum_{j=1}^{J} a_j\right] + \sum_{j=1}^{J}\left[\phi_{n,j}\,u_{n,j} + \psi_{n,j}\,v_{n,j}\right] \quad = \quad r_{\boldsymbol{\theta},n} \qquad \text{(A19)}$$

is equivalent to the target matrix equation

$$K\,\boldsymbol{z} \quad = \quad \boldsymbol{r_\theta} \qquad \text{(A20)}$$

if $r_{\boldsymbol{\theta},n}$ is the n-th element of the residual vector $\boldsymbol{r_\theta}$ defined in Section 2. Equation (A12) through Equation (A19) define a banded matrix equation in the "extended" space

$$K_{\rm ext}\,\boldsymbol{z}_{\rm ext} \quad = \quad \boldsymbol{r}_{\boldsymbol{\theta},\,\rm ext} \quad , \qquad \text{(A21)}$$

and, as before, this can be used to solve for $K^{-1}\,\boldsymbol{r_\theta}$ and $\det K$ in $\mathcal{O}(N)$ operations. Figure A1 shows a pictorial representation of the sparsity pattern of the extended matrix $K_{\rm ext}$. The extended matrix has the dimension $N_{\rm ext} = (4\,J+1)\,N - 4\,J$, with a bandwidth $W = 2\,J + 2$ and $N_{\rm nz} = (20\,J+1)\,N - 28\,J$ non-zero entries. Given

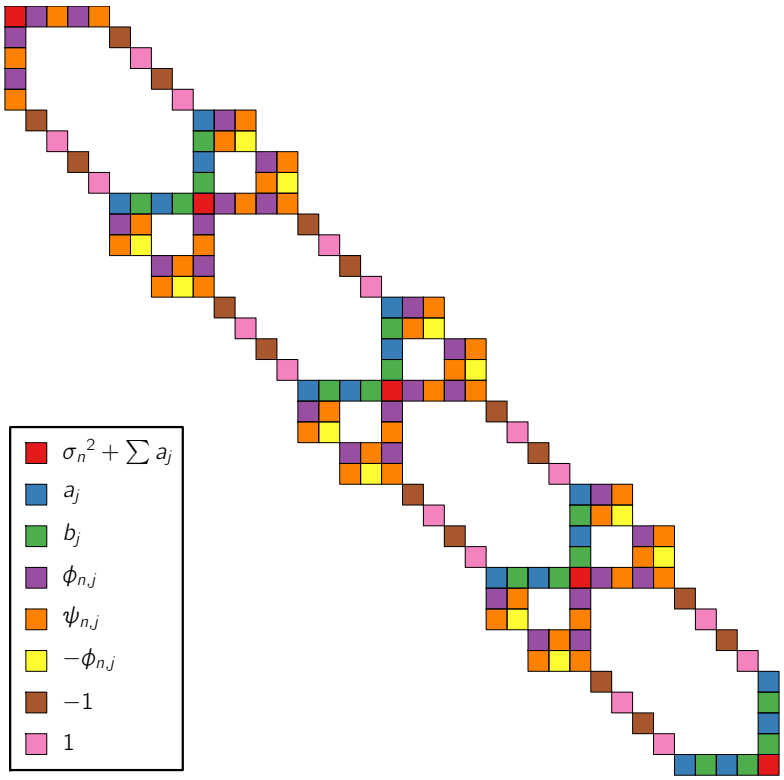

Figure A1. A pictorial representation of the sparse extended matrix $K_{\rm ext}$ (Equation A21) with $N = 5$ and $J = 2$. Each colored block corresponds to a non-zero entry in the matrix as described in the legend.

this definition of $K_{\rm ext}$, the corresponding extended vectors $\boldsymbol{z}_{\rm ext}$ and $\boldsymbol{r}_{\rm ext}$ are defined schematically as

$$\boldsymbol{z}_{\rm ext}{}^{\rm T} = \begin{pmatrix} z_1 & u_{1,j} & v_{1,j} & g_{1,j} & h_{1,j} & z_2 & u_{2,j} & \cdots & h_{N-1,j} & z_N \end{pmatrix} \tag{A22}$$

and

$$\boldsymbol{r}_{\rm ext}{}^{\rm T} = \begin{pmatrix} r_{\boldsymbol{\theta},1} & 0 & 0 & 0 & 0 & r_{\boldsymbol{\theta},2} & 0 & \cdots & 0 & r_{\boldsymbol{\theta},N} \end{pmatrix} \quad . \tag{A23}$$

After constructing the extended matrix (using a compact storage format), the extended matrix can be factorized using a LU-decomposition[6] routine optimized for band or sparse matrices. This decomposition can then be used to compute the determinant of K, solve $K^{-1}\,\boldsymbol{r}_{\boldsymbol{\theta}}$, and subsequently calculate the marginalized likelihood in Equation (3).

In practice, it is worth treating terms with $b_j = 0$ and $d_j = 0$ as a special case because, for this term j, $\psi_{n,j}$, $h_{n,j}$, and $v_{n,j}$ will also be identically zero for all n. Removing these trivial rows from the extended matrix results in factor of two increase in the computational efficiency for these terms. In our implementation, we refer to "real" terms as those where $b_j = 0$ and $d_j = 0$, and the general terms are called

[6] Even though $K_{\rm ext}$ is symmetric, it is not positive definite so a Cholesky solver cannot be used for increased efficiency.

"complex".

A.3. *Implementation considerations & scaling*

The extended system defined in the previous section is sparse, with typically fewer than a few percent non-zero entries and band structure. In this section, we empirically investigate the performance and scaling of three different algorithms for solving this extended system:

1. `vanilla`: A simple algorithm for computing the LU decomposition for banded matrices using Gaussian elimination (we implemented this method following Press et al. 1992; Press et al. 2007),
2. `lapack`: The general banded LU decomposition implementation from LAPACK[7] (Anderson et al. 1999) using optimized BLAS routines,[8] and
3. `sparse`: A general sparse LU solver – the `SparseLU` solver from Eigen (Guennebaud et al. 2010) – that exploits the sparsity but not the band structure.

The theoretical scaling for a band LU decomposition is $\mathcal{O}(N\,J^3)$ because the dimension of the extended matrix scales as $N\,J$ and the bandwidth scales with J (Press et al. 1992; Press et al. 2007). Ambikasaran (2015) found an empirical scaling of $\mathcal{O}(N\,J^2)$ that used the sparse LU decomposition implemented in the SuperLU package (Demmel et al. 1999). We find that, while the `vanilla` solver scales as expected, the `lapack` implementation scales empirically as $\mathcal{O}(N\,J^2)$ and offers the fastest solves for $J \gtrsim 8$ on all platforms that we tested.

The benchmark experiments shown here were performed on a MacBook Pro with two 2.6 GHz CPUs but we find similar results on a Dell workstation with 16 2.7 GHz CPUs and running Ubuntu. Figure A2 shows how the cost of computing Equation (3) scales with N and J using the `vanilla` solver. As expected theoretically, the scaling is linear in N for all N and cubic in J for large J. Figure **??** and Figure **??** are the same plots for the `lapack` and `sparse` solvers respectively. For each of these optimized solvers, the empirical scaling is $\mathcal{O}(N\,J^2)$ at the cost of some extra overhead. Therefore, for $J \lesssim 5$ or 10 the `vanilla` solver is more efficient than the other algorithms. The real world performance of celerite depends on the specific platform, hardware, and LAPACK/BLAS implementation, but we have found qualitatively similar results across popular platforms and state-of-the-art libraries.

B. ENSURING POSITIVE DEFINITENESS

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

[7] We use the `dgbtrf` and `dgbtrs` methods from LAPACK.

[8] The experiments in this paper use the Intel Math Kernel Library (MKL): `https://software.intel.com/en-us/intel-mkl`.

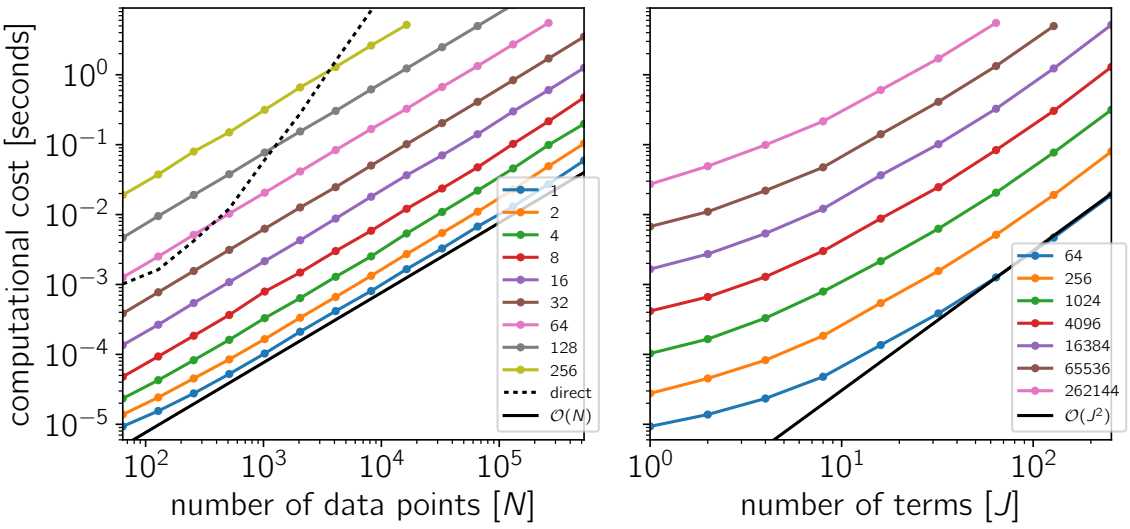

Figure A2. *DFM*: Cholesky A benchmark showing the computational scaling of celerite using the `vanilla` band solver as a function of the number of data points and the number of terms. *(left)* The cost of computing Equation (3) with a covariance matrix given by Equation (8) as a function of the number 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)* The 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.

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

$$\frac{(a_j\,c_j + b_j\,d_j)\,({c_j}^2 + {d_j}^2) + (a_j\,c_j - b_j\,d_j)\,\omega^2}{\omega^4 + 2\,({c_j}^2 - {d_j}^2)\,\omega^2 + ({c_j}^2 + {d_j}^2)^2} > 0 \quad . \tag{B24}$$

The denominator is positive for all $c_j \neq 0$ and it can be shown that, when $c_j = 0$, Equation (B24) 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

$$a_j\,c_j > -b_j\,d_j \tag{B25}$$

$$a_j\,c_j > b_j\,d_j \quad . \tag{B26}$$

Furthermore, we can see that a_j must be positive since $k(0) = a_j$ should should be positive and, similarly, by requiring the covariance to be finite at infinite lag, we obtain

the constraint $c_j \geq 0$. Combining these results, we find the constraint

$$|b_j\, d_j| < a_j\, c_j \quad . \tag{B27}$$

In the case of J *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 $z = \omega^2$:

$$S(\omega) = \sum_{j=1}^{J} \frac{q_j\, z + r_j}{z^2 + s_j\, z + t_j} = 0 \tag{B28}$$

where

$$\begin{aligned} q_j &= a_j\, c_j - b_j\, d_j & \text{(B29)} \\ r_j &= (d_j^2 + c_j^2)(b_j\, d_j + a_j\, c_j) & \text{(B30)} \\ s_j &= 2(c_j^2 - d_j^2) & \text{(B31)} \\ t_j &= (c_j^2 + d_j^2)^2. & \text{(B32)} \end{aligned}$$

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

$$Q_0(z) = \sum_{j=1}^{J} (q_j z + r_j) \prod_{k \neq j} \left(z^2 + s_k z + t_k\right) = 0 \quad , \tag{B33}$$

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 upon 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

$$Q_k(z) = -\mathrm{rem}(Q_{k-2},\, Q_{k-1}) \tag{B34}$$

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 polynomial 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 which 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 non-zero real root is found, the likelihood may be set to zero.

REFERENCES

Agol, E., Steffen, J., Sari, R., & Clarkson, W. 2005, Monthly Notices of the Royal Astronomical Society, 359, 567

Aigrain, S., Parviainen, H., & Pope, B. J. S. 2016, Monthly Notices of the Royal Astronomical Society, 706

Aigrain, S., Llama, J., Ceillier, T., et al. 2015, Monthly Notices of the Royal Astronomical Society, 450, 3211

Ambikasaran, S. 2015, Numer. Linear Algebra Appl., 22, 1102

Ambikasaran, S., Foreman-Mackey, D., Greengard, L., Hogg, D. W., & O'Neil, M. 2016, IEEE Transactions on Pattern Analysis and Machine Intelligence, 38, 252

Anderson, E., Bai, Z., Bischof, C., et al. 1999, LAPACK Users' Guide, 3rd edn. (Philadelphia, PA: Society for Industrial and Applied Mathematics)

Anderson, E. R., Duvall, Jr., T. L., & Jefferies, S. M. 1990, ApJ, 364, 699

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

Bond, J. R., Crittenden, R. G., Jaffe, A. H., & Knox, L. 1999, Comput. Sci. Eng., Vol. 1, No. 2, p. 21 - 35, 1, 21

Bond, J. R., & Efstathiou, G. 1987, Monthly Notices of the Royal Astronomical Society, 226, 655

Brewer, B. J., & Stello, D. 2009, Monthly Notices of the Royal Astronomical Society, 395, 2226

Brown, T. M. 2001, The Astrophysical Journal, 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, The Astrophysical Journal, 830, 138

Carter, J. A., & Winn, J. N. 2009, The Astrophysical Journal, 704, 51

Chaplin, W. J., Kjeldsen, H., Christensen-Dalsgaard, J., et al. 2011, Science, 332, 213

Chaplin, W. J., Basu, S., Huber, D., et al. 2013, The Astrophysical Journal Supplement Series, 210, 1

Claret, A., & Bloemen, S. 2011, Astronomy & Astrophysics, 529, A75

Corsaro, E., & Ridder, J. D. 2014, Astronomy & Astrophysics, 571, A71

Czekala, I., Mandel, K. S., Andrews, S. M., et al. 2017, ArXiv e-prints, arXiv:1702.05652

Dawson, R. I., Johnson, J. A., Fabrycky, D. C., et al. 2014, ApJ, 791, 89

Demmel, J. W., Eisenstat, S. C., Gilbert, J. R., Li, X. S., & Liu, J. W. H. 1999, SIAM J. Matrix Analysis and Applications, 20, 720

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, §24, (Dover Publications), 112–116

Dumusque, X., Boisse, I., & Santos, N. C. 2014, The Astrophysical Journal, 796, 132

Evans, T. M., Aigrain, S., Gibson, N., et al. 2015, Monthly Notices of the Royal Astronomical Society, 451, 680

Foreman-Mackey, D. 2016, The Journal of Open Source Software, 24, doi:10.21105/joss.00024

Foreman-Mackey, D., Agol, E., & Angus, R. 2017, dfm/celerite: celerite v0.1.3, , , doi:10.5281/zenodo.438359

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), , , doi:10.5281/zenodo.11989

Foreman-Mackey, D., & Morton, T. 2016, dfm/transit: v0.3.0, , , 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, Publications of the Astronomical Society of the Pacific, 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, Journal of The Korean Astronomical Society, 48, 93

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 ESA Special Publication, Vol. 235, Future Missions in Solar, Heliospheric & Space Plasma Physics, ed. E. Rolfe & B. Battrick

Haywood, R. D., Cameron, A. C., Queloz, D., et al. 2014, Monthly Notices of the Royal Astronomical Society, 443, 2517

Holman, M. J. 2005, Science, 307, 1288

Huber, D., Stello, D., Bedding, T. R., et al. 2009, Communications in Asteroseismology, 160, 74

Huber, D., Bedding, T. R., Stello, D., et al. 2011, ApJ, 743, 143

Hunter, J. D., et al. 2007, Computing in science and engineering, 9, 90
Jones, E., Oliphant, T., Peterson, P., et al. 2001, SciPy: Open source scientific tools for Python, ,
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
Kipping, D. M. 2013, MNRAS, 435, 2152
Knutson, H. A., Charbonneau, D., Allen, L. E., et al. 2007, Nature, 447, 183
Littlefair, S. P., Burningham, B., & Helling, C. 2017, MNRAS, 466, 4250
Loeb, A., & Gaudi, B. S. 2003, The Astrophysical Journal, 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 e-prints, arXiv:1702.05488
MacLeod, C. L., Ivezić, Ž., Kochanek, C. S., et al. 2010, ApJ, 721, 1014
Mandel, K., & Agol, E. 2002, The Astrophysical Journal, 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, Monthly Notices of the Royal Astronomical Society, 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 (Springer)
Pancoast, A., Brewer, B. J., Treu, T., et al. 2014, Monthly Notices of the Royal Astronomical Society, 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
Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. 1992, Numerical recipes in FORTRAN. The art of scientific computing (Cambridge University Press)
Press, W. H., Teukolsky, S. A., Vetterling, W. T., & Flannery, B. P. 2007, Numerical Recipes 3rd Edition: The Art of Scientific Computing (Cambridge University Press)
Rajpaul, V., Aigrain, S., Osborne, M. A., Reece, S., & Roberts, S. 2015, Monthly Notices of the Royal Astronomical Society, 452, 2269
Rasmussen, C. E., & Williams, K. I. 2006, Gaussian Processes for Machine Learning (MIT Press)
Rauer, H., Catala, C., Aerts, C., et al. 2014, Experimental Astronomy, 38, 249
Ricker, G. R., Winn, J. N., Vanderspek, R., et al. 2014, in Space Telescopes and Instrumentation 2014: Optical, Infrared, and Millimeter Wave, ed. J. M. Oschmann, M. Clampin, G. G. Fazio, & H. A. MacEwen (SPIE-Intl Soc Optical Eng)
Rybicki, G. B., & Press, W. H. 1992, ApJ, 398, 169
—. 1995, Physical Review Letters, 74, 1060
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
Stello, D., Cantiello, M., Fuller, J., et al. 2016, Nature, 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, The Astrophysical Journal, 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, Physical review, 36, 823
Uttley, P., McHardy, I. M., & Vaughan, S. 2005, Monthly Notices of the Royal Astronomical Society, 359, 345
Van Der Walt, S., Colbert, S. C., & Varoquaux, G. 2011, Computing in Science & Engineering, 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, University of Cambridge
Wilson, A. G., & Adams, R. P. 2013, in ICML (3), 1067–1075
Wilson, A. G., Dann, C., & Nickisch, H. 2015, arXiv preprint arXiv:1511.01870, `http://arxiv.org/abs/1511.01870`
Wilson, A. G., & Nicisch, H. 2015, Proceedings of the 32nd International Conference on Machine Learning, 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 e-prints, arXiv:1612.04834
Zu, Y., Kochanek, C. S., & Peterson, B. M. 2011, ApJ, 735, 80
Zucker, S., Mazeh, T., & Alexander, T. 2007, The Astrophysical Journal, 670, 1326