



The
University
Of
Sheffield.

Department
Of
Mechanical
Engineering

MEng Mechanical Engineering

Identifying Dynamic Systems with Probabilistic Numerics

Harvey WALTON

May 2024

Supervisor: Dr. Timothy J. Rogers

Thesis submitted to the University of Sheffield in partial
fulfilment of the requirements for the degree of Master of
Engineering

Word count: 7946

Summary

In mechanical engineering, the field of dynamics studies how systems, like structures and fluids, respond and evolve over time under varying forces. Being able to characterise these systems by identifying their dynamic properties and parameters helps to design structures that last longer before failing from fatigue, and that are less likely to fail structurally from excessive excitation under external forces. The identification of dynamic systems often uses the “Frequency Response Function”, which relies on the “Fourier transform” to convert data from a function of time to a function of frequency. This paper explores a new paradigm of “probabilistic numerics” over traditional numerical methods that allows the Fourier transform to be performed through the use of a Gaussian Process (GP) machine learning model. The model showed significantly better robustness to noise compared to the industry standard Fast Fourier Transform (FFT), exhibiting an improved Mean Squared Error in magnitude versus the true analytical Fourier transform by a factor of 9.66. Therefore, it could allow dynamic systems to be identified more accurately in across industry where data is noisy, from aviation to renewable power generation. Furthermore, the model had several promising features that made it more versatile in this context. Most notably, the fact that the time domain input data need not be evenly spaced enabled it to detect frequencies that would be above the Nyquist limit of what would be detectable by an equivalent FFT of the same number of datapoints sampled evenly across the same time period. This makes the model most suitable in industrial contexts where there are constraints to the amount of data that can be collected due to cost, availability, or limited memory. However, the computational speed of this GP model is currently orders of magnitude slower than the FFT, so before it can be a viable alternative in practice, it would need development to allow it to scale better to large datasets.

Nomenclature

Greek Symbols

δ	The Dirac delta function
ϵ_j	The noise in observation, normally distributed with a mean of zero and a variance of σ_y^2
ζ	The damping ratio
θ	The set of hyperparameters
$\mu(t)$	The mean function
$\mu_X \mu_*$	Vectors containing the prior mean of the distribution of values of the latent function, at the training points, \mathbf{X} , and test points \mathbf{X}_* respectively

$\mu_{* \mathbf{X}}$	The posterior mean of the Gaussian Process, given the training data, \mathbf{X}
\mathbf{M}_*	The vector containing the prior mean of the frequency domain values of the latent function at test points, ξ
ξ	The vector of frequency domain test points ξ
ξ	The angular frequency
$\xi_j \ \xi_k$	The j^{th} and k^{th} elements in ξ
ξ_r	The resonant frequency
σ^2	The variance hyperparameter of the Squared Exponential Kernel
σ_y^2	The variance of observation noise
$\Sigma(t, t)$	The covariance function of the Gaussian Process in the time domain
$\Sigma_{* \mathbf{X}}$	The posterior covariance of the Gaussian Process, given the training data, \mathbf{X}

Latin Symbols

\mathbf{A}, \mathbf{B}	Vectors with elements a, b - subscript indicates index of element
a, b	Elements of vectors \mathbf{A}, \mathbf{B} respectively
c	The damping coefficient
\mathbf{D}	The training data
\mathbf{F}_*	The vector of latent function values in the frequency domain at a set of test points ξ
$f(x)$	The latent function of the variable x
\mathbf{f}_*	The vector of latent function values at a set of test points \mathbf{X}_*
FFT	Fast Fourier Transform
$\mathcal{F}_p[f](\xi)$	Fourier transform of $f(t)$, with respect to p
GP	Gaussian Process
\mathbf{I}	The identity matrix
$\mathbf{K}_{X,X}$	A covariance matrix defined by $k(\mathbf{X}, \mathbf{X})$
$\mathbf{K}_{X,*}$	A covariance matrix defined by $k(\mathbf{X}, \mathbf{X}_*)$

$\mathbf{K}_{*,*}$	A covariance matrix defined by $k(\mathbf{X}_*, \mathbf{X}_*)$
\mathbf{K}_σ	A covariance matrix with noise term, $\mathbf{K}_{X,X} + \sigma_y^2 \mathbf{I}$
$\mathcal{K}_{X,\xi}$	A covariance matrix with j^{th} element defined by $\mathcal{F}_{\xi_k}[k(t_j, \xi_k)]$
$\mathcal{K}_{\xi,\xi}$	A covariance matrix with j^{th} element defined by $\mathcal{F}_{\xi_j}[\mathcal{F}_{\xi_k}[k(\xi_j, \xi_k)]]$
$\mathcal{K}(\xi, 0)$	The Fourier transform of the (Squared Exponential) kernel function evaluated at ξ
k	The stiffness
$k(a, b)$	The kernel function, evaluating the similarity of a and b
l	The length scale hyperparameter of the Squared Exponential Kernel
M	The number of inducing points in Sparse GP
m	The mass
N	The number of training or test points
NLML	Negative Log Marginal Likelihood
$O()$	Big O notation, used to describe the complexity - or “Order” of an algorithm
\mathbf{X}	A vector of training point inputs
\mathbf{X}_*	A vector of test point inputs
$t_j \ t_k$	The j^{th} and k^{th} elements in the vector of training points \mathbf{X}
\mathbf{y}	The vector of y_j values from the set of training points
y_j	The noisy observed output based on the latent function, $f(x)$

Contents

1	Background and Understanding of the Problem	1
1.1	Dynamic Systems	1
1.2	Identifying Dynamic Systems	2
1.3	Classically Quantifying Uncertainty with Error Bounds	3
1.4	The Fourier Transform and the Frequency Response Function (FRF)	4
1.5	The Fast Fourier Transform Algorithm	4
1.6	Probabilistic Numerics and the Gaussian Process	6
2	Aims and Objectives	8
2.1	Aim	8
2.2	Objectives	8
3	Creating the Gaussian Process Model	9
3.1	The Data	9
3.2	The Kernel	9
3.3	Hyperparameter Optimisation	10
3.4	The Prediction Function	11
4	Adjusting the Model for the Frequency Domain	11
4.1	Finding the Posterior Mean of the Fourier Transform	11
4.2	Finding the Posterior Covariance of the Fourier Transform	12
5	Results and Discussion	13
5.1	The Basic Dataset	13
5.1.1	Accuracy of the Fourier Transform	13
5.2	Robustness to Noise	16
5.3	Non-Uniform Sampling of Frequency Domain	16
5.4	Non-Uniform Sampling of the Time Domain and Overcoming Aliasing and the Nyquist Limit	18
5.5	The Posterior Covariance of the Fourier Transform	19
6	Directions for Future Work	21
6.1	Alternative Kernels	21
6.2	Computation Time and Scaling to Large Data	21
7	Conclusion	22
	References	24

Appendices	25
A.1 The Use of Generative AI (ChatGPT)	25

1 Background and Understanding of the Problem

1.1 Dynamic Systems

In mechanical engineering, the field of dynamics studies how systems, like structures and fluids, respond and evolve over time under varying forces. These forces can range from aerodynamic pressures to impacts from waves or road irregularities. Excessive vibration or oscillation of machinery is known to reduce service life and in extreme cases cause catastrophic failure. Therefore, by studying these dynamic systems they could be designed to perform better, be safer and to have lower maintenance costs. However, the field of dynamics is far from being mastered since there are many types of problems that cannot yet be solved.

To begin with, here are some examples of dynamic systems:

1. Firstly, a motorcycle suspension system is a combination of springs, dampers and linkages that connect the motorcycle chassis to its wheels. Its primary purpose is to keep the tyres in contact with the ground, and without proper suspension, the tyres would lose traction when encountering bumps especially when braking, accelerating or cornering [1]. It is designed to allow some relative motion between the chassis and the wheels in order to absorb impacts while at the same time trying to maintain its equilibrium position, and zero velocity (both relative to the chassis).

The dampers in the suspension produce a force proportional to the velocity that acts to slow the wheel down. Viscous dampers do this by pushing a liquid through a small orifice as the wheel moves. Damping can be optimised (or made “critical”) by bringing the system back to equilibrium after a disturbance as fast as possible. Too little damping, and the wheel will overshoot and oscillate back and forth before reaching equilibrium. Too much damping, and although the wheel will not oscillate, it will reach equilibrium more slowly than optimal.

Similarly, the springs create stiffness in the system that produces a force with a magnitude proportional to the displacement of wheel which acts to return the wheel to its equilibrium position. Too high of a stiffness means that the system will not have enough time to absorb the impact softly, but too low means that the bike’s responses to steering inputs are slow and unreactive.

Studying this dynamic system could allow for the prediction of behaviour based on varying these parameters. This would enable optimisation without trial and error, conserving resources during the research and development phase of the product lifecycle.

2. Secondly, offshore wind turbines are known to experience high cycle fatigue due to a relentless random forcing the waves that repeatedly crash against the base of its tower. This is a phenomenon that causes any micro-defects to grow into cracks over time due to cyclic

loading below the elastic limit of the material, which reduces the life of the structure due to increased risk of mechanical failure.

The rate at which this occurs increases with the amplitude of the stresses in the structure, which in turn increase with the amplitude of oscillation in response to this cyclic loading. If a periodic force is applied to any system with a specific frequency known as a resonant frequency, the amplitude of the oscillation may become very large even for a small amplitude force, leading to an excessive rate of high cycle fatigue. The same can happen in the case of the wind turbine that receives a random force with an *average* frequency similar to this resonant frequency. It is even possible for the stresses in a structure to become so large that they exceed the ultimate tensile strength of the material, causing a sudden mechanical failure and the collapse of the turbine.

Therefore, to make structures safer and reduce maintenance costs, it would be valuable to be able to model their dynamic properties (such as this resonant frequency) from system parameters (such as the stiffness of the tower) so that the frequencies at which they occur can be designed away from the expected frequencies of the periodic forces.

3. Finally, an aircraft wing bends in response to aerodynamic forces. However, the forces themselves change as the wing twists and bends. The interdependence of these two effects creates complicated interactions that are difficult to model accurately. But it is important to do so because if at high airspeed, energy from the wing forcing is added to the system faster than it can be dissipated, the positive feedback loop can lead to a phenomenon called aerodynamic flutter where the oscillations become larger and larger until they cause catastrophic failure.

1.2 Identifying Dynamic Systems

So, dynamic systems have parameters such as mass (m), damping coefficient (c) and stiffness (k) which are fundamental characteristics of the system. “Forwards modelling” is when the observable behaviour of the system, known as the system properties, are predicted from the system parameters using physical laws. These properties include the resonant frequencies which are the forcing frequencies that produce a peak in amplitude of excitation, the mode shapes which are the shapes of oscillations at peaks in systems with multiple degrees of freedom, and the damping ratios which compare the damping coefficient to that of critical damping at each resonant frequency.

Often a distinction is made between linear and nonlinear systems. Only linear systems comply with the principle of superposition, which states that “when two or more waves simultaneously pass through a point, the disturbance at the point is given by the sum of the disturbances each wave would produce in absence of the other waves” [2]. This means that if you double the input force over time, the response over time is exactly double, but otherwise identical. This is not the

case for nonlinear systems (such as the aircraft wing in the example above) where non-compliance with the principle of superposition causes the system properties to distort as the magnitude of the input force is increased. This, in turn, causes the response to become highly complex and chaotic, meaning that it is highly sensitive to initial conditions. This is why forward modelling is currently too difficult for systems that are non-linear.

Unfortunately, everything is non-linear in real life. Only in rare cases are systems close enough to being linear that they can be approximated as such. What's worse is that even if a model did exist to solve the nonlinear system, it would be of no use because typically not even the system parameters are known to be able to input them into the model. This is why the field of dynamics has not been mastered, and all that can currently be done for medium to highly non-linear systems - *which is almost all systems* - is to take real life data measurements from an existing system and identify the system properties and parameters that best fit the observed data through what is known as "inverse modelling". However, even this process is fraught with issues and uncertainty.

1.3 Classically Quantifying Uncertainty with Error Bounds

Identifying system properties and parameters using this inverse modelling approach results in values that are not exact, but have an uncertainty associated with them that comes from multiple sources. The first source of uncertainty is noise that is always present to some degree in sampled input data due to the measuring devices - typically accelerometers - that are used to capture the force input and corresponding response from the system. Furthermore, even if the true input and response could be sampled exactly, the non-identifiability problem states that since there are multiple sets of system parameters that could give the same set of observed outputs, it's impossible to uniquely determine them [3]. Similarly, many algorithms make use of mathematical objects (such as matrix pseudoinverses) which have multiple alternatives - any of which could be used - where the correct choice is subjective. This noise and lack of uniqueness is reflected in the outputted system properties and parameters as uncertainty.

Time passes continuously forever, and the governing equations of dynamic systems are defined mathematically in the continuous domain. In theory, these could be solved "analytically" to give an exact mathematical solution in the continuous domain, but in practice, data sampled from the real world is usually discrete, meaning that it is defined by a finite set of points separated by finite time intervals. In addition, many sets of governing equations are currently too complex to solve analytically for systems that are not highly idealised. This is why when modelling, problems are classically solved using "numerical methods", which are discrete approximations of analytical solutions of the governing equations. Here, uncertainty of values is often quantified using error bounds derived from sensitivity analysis. For instance, the error bound of a stiffness could be represented as $0.1 \pm 0.02 \text{ Nm}^{-1}$, indicating the expected range of the true value. This process involves estimating the uncertainty of input data, usually from the measuring equipment's

resolution or variability in sample data. From these estimates, the upper and lower bounds of the output parameters can be calculated using the upper and lower bounds of the input parameters to give the uncertainty. However, this is not straightforward and typically requires additional techniques and considerations beyond the core algorithm.

1.4 The Fourier Transform and the Frequency Response Function (FRF)

When inverse modelling during the analysis of dynamic systems, any collected time series data is almost always converted into the frequency domain, meaning the output becomes a function of frequency instead of a function of time. This is because it makes it easier to calculate the system properties since it gives their physical meaning a clear visual interpretation, and is done using a linear operator called the Fourier transform. Specifically, the Frequency Response Function (FRF) is computed, which is the ratio between the Fourier transforms of the response and input. This function is complex, which means it has two components that provide information regarding the amplitude (magnitude) and phase (lag) of the response versus the input. If the system parameters (mass, damping coefficient or stiffness) change, so do the system properties. This is visible in the FRF in Figure 1.1, where a larger resonant frequency creates a peak in excitation at a higher frequency, and a lower damping ratio results in a sharper peak. Note the phase shifts by $\pi \text{ rad}$ (180°) across a resonant frequency.

The Fourier transform, $\mathcal{F}(\xi)$, is defined in Equation 1.1

$$\mathcal{F}[f](\xi) = \frac{1}{2\pi} \int_{-\infty}^{\infty} f(t)e^{-i\xi t} dt \quad 1.1$$

where ξ is the angular frequency [rad s^{-1}] and $f(t)$ is the time domain latent function being operated on. This is an extremely powerful analytical technique that is at the core of signal processing, which is a field widely used in engineering to analyse, manipulate and interpret experimental data that changes over time in order to extract useful information or improve them for a specific purpose. However, for this to be applied in its base form to a function, it must be defined for every point in time in the continuous domain $(-\infty, \infty)$. Therefore, when working with sampled real world data, an approximation of this function is classically found using a numerical method called a Discrete Fourier Transform (DFT), specifically the Fast Fourier Transform (FFT).

1.5 The Fast Fourier Transform Algorithm

Although discovering this fast way to compute the DFT revolutionised many engineering disciplines such as structural health monitoring, image compression, signal processing, and control theory [4], its implementation has a number of challenges that have to be carefully handled. Firstly,

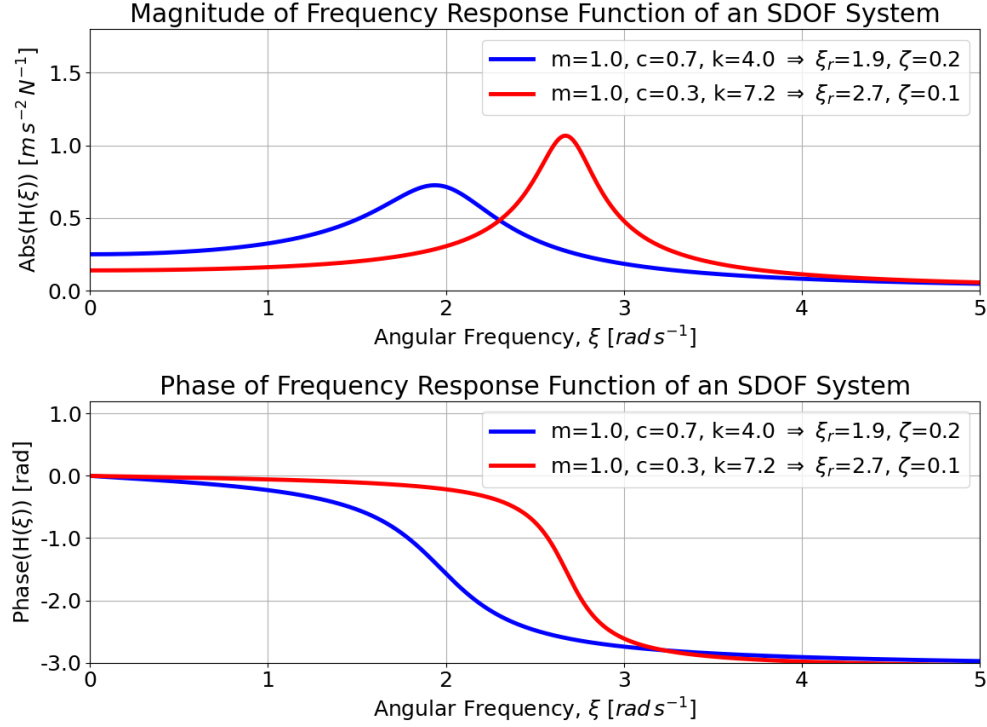


Figure 1.1 - The magnitude and phase of $H(\xi)$, a frequency response function of a single degree of freedom (SDOF) system for an acceleration measured in response to an input force, showing how changing its system parameters mass m [kg], damping coefficient c [Ns m⁻¹] and stiffness k [Nm⁻¹] affects the system parameters resonant frequency ξ_r [rad s⁻¹] and damping ratio ζ [-].

the frequency domain is both positive and negative, resulting in symmetry where the energy of the system is split evenly about zero. Secondly, the FFT assumes the input signal is finite in length and periodic, which can lead to a phenomenon called spectral leakage if the signal is not an exact multiple of the chosen total time period [5]. This can cause sharp resonance peaks to be smoothed out in the output because energy that should be concentrated at a peak frequency is spread across a range of frequencies nearby, and is typically mitigated by using a window such as a Hanning window to decrease the magnitude of the time domain signal at either end to smoothly bring it to zero at the boundary between the end and start, thus minimizing and abrupt changes that could lead to distortion of the spectrum. Also, the FFT can be highly sensitive to noise in the data, meaning a small amount of noise can create large fluctuations in the shape of the frequency spectrum, masking the true underlying frequencies and modal properties [5]. Similarly, since the time domain and resulting frequency domain data is discrete, the basic algorithm can exhibit large variability in the amplitude and frequency of peaks depending on if the sampled frequencies fall exactly on the peak or either side of it, which in turn can affect modal properties and parameters calculated downstream of this data in the context of dynamics. Finally, there is a highest “Nyquist” frequency that is half the sampling frequency, for which any higher frequencies present in the time domain signal will cause aliasing. This is where frequencies that are too high are reflected back at this Nyquist frequency.

However, over time this spectra will need to be measured more accurately and at higher frequencies in the engineering industry. For example, as scramjet engines are developed in the aerospace industry for faster hypersonic aircraft, the vibration endured will be stronger and at higher frequencies than in engines available today. This can lead to failure of components from fatigue, so being able to accurately measure the FFT spectra at these higher frequencies allows engineers to design effective structures that can survive the design life of the vehicle while minimising weight of material.

Currently, if spectra from the FFT need to be more accurate and be measured at higher frequencies, this is done by simply collecting more and more data, but this is not always feasible since there are limits to the available resources such as time, energy and capital required to collect this data. For the scramjet engine, hypersonic wind tunnel tests only last on the order of milliseconds [6] to reduce the cost of energy used and to prevent damage to the engine from the resulting high temperatures. This provides very little time for vibration data to be collected. Similarly, if measurement instruments with higher sampling frequencies are required to bring the limiting Nyquist frequency above the high frequency vibrations, the cost of this equipment and the computation required to process the data will increase.

1.6 Probabilistic Numerics and the Gaussian Process

This paper explores the potential advantages of a new paradigm of “probabilistic numerics” over traditional numerical methods that may allow one to identify dynamic systems using the Fourier transform more accurately from less data. Here, probabilistic reasoning is incorporated into the process of computation by treating “a numerical problem as one of statistical inference instead” [7]. The uncertainty of outputs is modelled with a probability distribution as opposed to the error bounds as in traditional numerical methods, which may improve robustness to noise, among other things. This paper does this through the implementation of a promising type of machine learning model called a Gaussian Process (GP) to compute the Fourier transform.

A GP uses noisy, discrete sampled data to model the probability distribution of a continuous “latent” function, $f(x)$, that represents the underlying relationship between an input, x , and an output, $f(x)$. In essence, it is “a collection of random variables, any finite number of which have a joint Gaussian distribution” [8] which describes how the probability distribution of each variable depends on the other variables. These variables are outputs of the latent function at a specific input point and include both the training data (noisy observed samples at points in time), and the prediction points at which the true value is unknown.

Assuming a mean of the latent function of zero, and a noisy observed force, $y_j = f(x_j) + \epsilon_j$, where $\epsilon_j \sim \mathcal{N}(0, \sigma_y^2)$, this joint distribution of the observed data and the latent, noise-free function at the test points is given [9] by Equation 1.2

$$\begin{Bmatrix} \mathbf{y} \\ \mathbf{f}_* \end{Bmatrix} \sim \mathcal{N} \left(\begin{Bmatrix} \boldsymbol{\mu}_X \\ \boldsymbol{\mu}_* \end{Bmatrix}, \begin{bmatrix} \mathbf{K}_{X,X} + \sigma_y^2 \mathbf{I} & \mathbf{K}_{X,*} \\ \mathbf{K}_{X,*}^T & \mathbf{K}_{*,*} \end{bmatrix} \right) \quad 1.2$$

where ϵ_j is the observational noise; \mathbf{y} is a vector of y_j values from a set of training points; $\boldsymbol{\mu}_X$ and $\boldsymbol{\mu}_*$ are vectors containing the prior mean of the distribution of values of the latent function, at the training and test points, \mathbf{X} and \mathbf{X}_* respectively; \mathbf{f}_* is a vector of function values at a set of test points \mathbf{X}_* ; \mathbf{I} is the identity matrix and covariance matrices $\mathbf{K}_{X,X}$, $\mathbf{K}_{X,*}$ and $\mathbf{K}_{*,*}$ are defined by $k(\mathbf{X}, \mathbf{X})$, $k(\mathbf{X}, \mathbf{X}_*)$ and $k(\mathbf{X}_*, \mathbf{X}_*)$ respectively. Note that block matrix notation is used where matrices are conjoined to form a larger block matrix.

The kernel is type of function that measures the similarity between each pair of inputs based on a set of hyperparameters and is chosen to reflect any prior knowledge about the problem domain.

These hyperparameters need to be optimised by minimizing the Negative Log Marginal Likelihood (NLML), as defined [9] in Equation 1.3

$$-\log p(\mathbf{D}|\boldsymbol{\theta}) = \frac{1}{2}(\mathbf{y} - \boldsymbol{\mu}_X)^T \mathbf{K}_\sigma^{-1}(\mathbf{y} - \boldsymbol{\mu}_X) + \frac{1}{2} \log |\mathbf{K}_\sigma| + \frac{N}{2} \log(2\pi) \quad 1.3$$

where \mathbf{D} is the training data; N is the number of training points; $\boldsymbol{\theta}$ is the set of hyperparameters and $\mathbf{K}_\sigma = \mathbf{K}_{X,X} + \sigma_y^2 \mathbf{I}$ is the covariance matrix with the noise term.

The NLML measures how well the model explains the observed data for a given set of hyperparameters while penalizing complexity to avoid overfitting, which is when the model fits the training data so well that it generalises poorly to new test data. This can happen when the model has not only learnt the underlying function but also its associated noise, which will not be consistent between datasets due to its random nature. Optimisation in this manor works by identifying the hyperparameter values that are most likely to produce the observed training data.

To predict the latent function at one or multiple points, the ‘‘posterior’’ probability distribution of the function is determined by updating the initial prior assumptions using the training data. This process is called conditioning the joint Gaussian prior distribution on the observations [8], as shown in Equation 1.4, where $\boldsymbol{\mu}_{*|\mathbf{X}}$ and $\boldsymbol{\Sigma}_{*|\mathbf{X}}$ are the posterior mean and covariance of the GP, given the training data, \mathbf{X} as shown in Equation 1.5 and Equation 1.6 respectively.

$$p(\mathbf{f}^*|\mathbf{D}, \mathbf{X}^*) = \mathcal{N}(\mathbf{f}^*|\boldsymbol{\mu}_{*|\mathbf{X}}, \boldsymbol{\Sigma}_{*|\mathbf{X}}) \quad 1.4$$

$$\boldsymbol{\mu}_{*|\mathbf{X}} = \boldsymbol{\mu}_* + \mathbf{K}_{X,*}^T (\mathbf{K}_{X,X} + \sigma_y^2 \mathbf{I})^{-1} (\mathbf{y} - \boldsymbol{\mu}_X) \quad 1.5$$

$$\boldsymbol{\Sigma}_{*|\mathbf{X}} = \mathbf{K}_{*,*} - \mathbf{K}_{X,*}^T (\mathbf{K}_{X,X} + \sigma_y^2 \mathbf{I})^{-1} \mathbf{K}_{X,*} \quad 1.6$$

Note that this GP is closed under the action of linear operators which means that it can be described using only a combination of operations that comply with the principle of superposition. It is also defined in the continuous domain $(-\infty, \infty)$, so the Fourier transform can be taken analytically without needing to resort to the discrete FFT. Incredibly, since the Fourier transform is a linear operator [10], the result of this operation on the GP is yet another GP. The joint distribution of the two can therefore be written as shown in Equation 1.7

$$\begin{Bmatrix} \mathbf{y} \\ \mathbf{F}_* \end{Bmatrix} \sim \mathcal{N} \left(\begin{Bmatrix} \boldsymbol{\mu}_X \\ \mathbf{M}_* \end{Bmatrix}, \begin{bmatrix} \mathbf{K}_{X,X} + \sigma_y^2 \mathbf{I} & \boldsymbol{\kappa}_{X,\xi} \\ \boldsymbol{\kappa}_{X,\xi}^T & \boldsymbol{\kappa}_{\xi,\xi} \end{bmatrix} \right) \quad 1.7$$

where \mathbf{F}_* is the frequency domain prediction of the latent function; \mathbf{M}_* is the vector containing the prior mean of the frequency domain values of the latent function at test points, ξ ; the matrix element $[\mathcal{K}_{x,\xi}]_{jk}$ denotes $\mathcal{F}_{\xi_k}[k(t_j, \xi_k)]$; the matrix element $[\mathcal{K}_{\xi,\xi}]_{jk}$ denotes $\mathcal{F}_{\xi_j}[\mathcal{F}_{\xi_k}[k(\xi_j, \xi_k)]]$; t_j is the j^{th} element in the vector of training points \mathbf{X} ; and ξ_j and ξ_k are the j^{th} and k^{th} elements of ξ . Like before, this distribution can be conditioned to find the posterior mean and covariance of the Fourier transform at a specific set of desired frequencies, ξ , given the training data, \mathbf{X} .

Modelling signals in this way provides an inherent mechanism to account for uncertainty and allows this information to be preserved naturally in the covariance matrix when performing linear operations on the data. Furthermore, the noise term $\epsilon_j \sim \mathcal{N}(0, \sigma_y^2)$ explicitly models the observational noise, giving the GP a way to distinguish between the signal and noise components. This means that the use of probabilistic numerics may allow better identification of dynamic systems since it may allow for the Fourier transform to be performed with better robustness to noise.

2 Aims and Objectives

2.1 Aim

The main aim of this paper is to improve the noise resilience of the Fourier transform used to convert data to the frequency domain.

2.2 Objectives

This overarching aim can be broken down into four objectives:

1. Create a working Gaussian Process model, which involves:
 - Sourcing some time series signal data.
 - Choosing and creating a kernel.

- Implementing optimisation of kernel hyperparameters.
 - Creating a prediction function to predict the outputs of new inputs.
2. Adjust this model to allow predictions of the mean and covariance functions in the frequency domain by finding their Fourier transforms in closed form.
 3. Test the robustness to noise of this model compared to the conventional Fast Fourier Transform in the context of a dynamic system.
 4. Explore any other quantitative and qualitative advantages and disadvantages of the model.

3 Creating the Gaussian Process Model

The first objective was completed by implementing the Gaussian Process (GP) model in the programming language Python.

3.1 The Data

Initially, training data was required to test the fit of the GP model during its creation. For this, the acceleration in the time domain of a single degree of freedom (SDOF) system in free vibration with system parameters $m = 1 \text{ kg}$; $c = 0.3 \text{ Nsm}^{-1}$; $k = 100 \text{ Nm}^{-1}$ was used. If this data was collected experimentally, some noise would always be present, so instead it was simulated. This allowed the levels of noise to be reduced to zero, as shown in the plot of “Training Data” in Figure 3.1, to allow for a broader range of test cases.

3.2 The Kernel

A Squared Exponential Kernel [11] was chosen to reflect the nature of the sample data, both of which are continuous and smooth, with no sudden jumps. In addition, the kernel has only two hyperparameters. This low number speeds up the hyperparameter optimisation process compared to alternative kernels. This function is defined in Equation 3.1

$$k(a, b) = \sigma^2 \exp \left(-\frac{(a - b)^2}{2l^2} \right) \quad 3.1$$

where $k(a, b)$ is the kernel function evaluated at elements a_i, b_j of vectors \mathbf{A}, \mathbf{B} respectively; σ^2 is the variance hyperparameter and l is the length scale parameter.

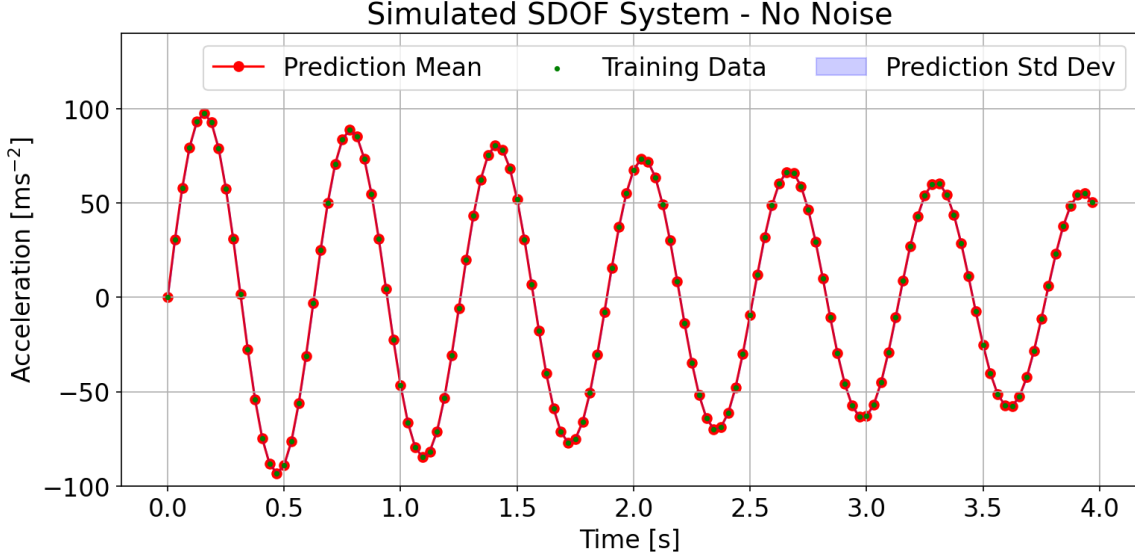


Figure 3.1 - The basic training dataset of time domain acceleration of a simulated SDOF system consisting of 128 datapoints at a sample rate of 32 Hz using system parameters $m = 1\text{ kg}$; $c = 0.3\text{ Nsm}^{-1}$; $k = 100\text{ Nm}^{-1}$ with no noise added, and the time domain prediction functions of the GP model fit to this data.

3.3 Hyperparameter Optimisation

An algorithm was designed to optimise the hyperparameters of the kernel by minimising the Negative Log Marginal Likelihood (NLML), for which a method was written (see Appendix. A.1). It did this by modifying each hyperparameter randomly in turn and seeing if the NLML improved. This was repeated for up to 100 iterations.

Imagine if the hyperparameter was only updated if the new NLML was lower than the previous. Since the shape of a cost function can be complex and highly dimensional, the algorithm could have got stuck in local minima and have never found the global minimum. This would be analogous to a ball that “tried” to find the bottom of a valley by rolling directly downhill; if it got stuck in a small ditch half-way down the larger hill, it would never find the lowest point.

Therefore, a mechanism was added to the algorithm to avoid this: If the new NLML was higher (i.e., worse) than the previous, the hyperparameter was maybe still updated, with a larger probability the closer the NLML was to the previous NLML. When the last iteration was complete, the lowest NLML of all the combinations hyperparameters was selected. This allowed the algorithm to explore after reaching a minimum to potentially find another part of the NLML function that was lower.

3.4 The Prediction Function

The prediction function was constructed, which accurately calculated the mean and standard deviation of the latent function as shown by the “Prediction Mean” and “Prediction Std Dev” plots in Figure 3.1 respectively. The mean of each prediction point is calculated using Equation 1.5 directly, whereas the standard deviation of each point uses only square root of the main diagonal of the covariance matrix resulting from Equation 1.6. In combination, this prediction function defines the probability distribution of the value of the latent function at any point, given the training data. Hence, when zero mean Gaussian noise with a standard deviation of 25 ms^{-2} was added to the training data, the standard deviation of the prediction became larger to take account for the uncertainty created by this noise, as shown in Figure 3.2.

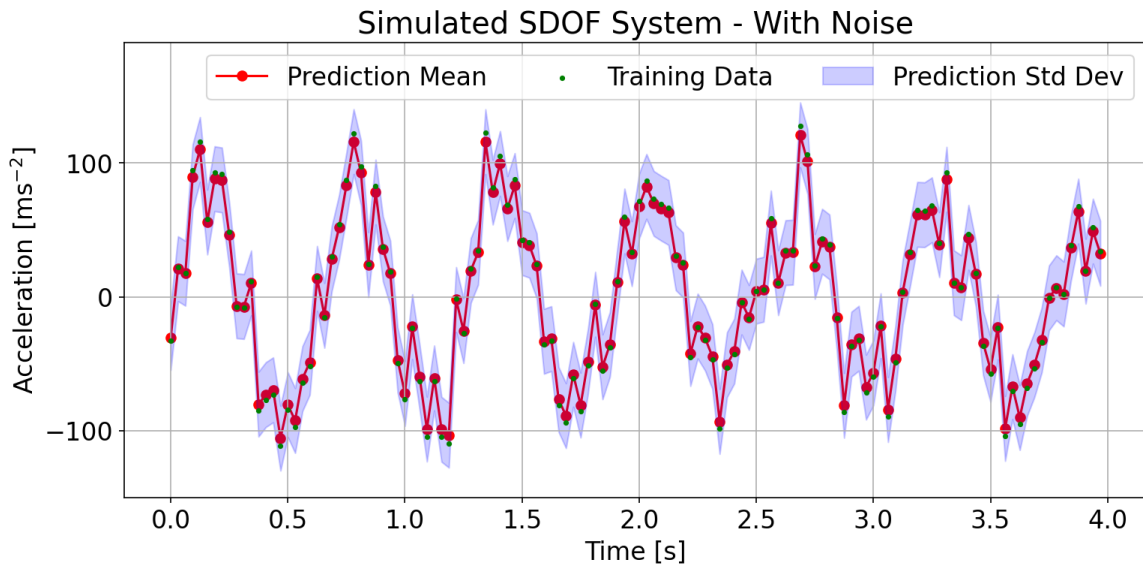


Figure 3.2 - The mean and standard deviation of the prediction function of the GP when noise was added to the training data, showing that the standard deviation of the prediction increases.

4 Adjusting the Model for the Frequency Domain

4.1 Finding the Posterior Mean of the Fourier Transform

Next, the function for the posterior mean of the Fourier transform was constructed, which could be evaluated at any set of points in the domain $(-\infty, \infty)$. This was done using a neat method described by Ambrogioni et al [12] which is equivalent to conditioning the joint distribution in Equation 1.7, but instead takes the Fourier transform of the time domain predictive mean function directly. Remarkably, this involved taking the Fourier transform of only the kernel, provided that

the Fourier shift theorem was first applied to ensure that the kernel was a function of a single variable, as shown in Equation 4.1

$$\mathcal{K}(\xi, 0) = \frac{\sigma^2 l}{\sqrt{2\pi}} \exp\left(-\frac{l^2 \xi^2}{2}\right) \quad 4.1$$

where ξ is the frequency and $\mathcal{K}(\xi, 0)$ is the Fourier transform of the Squared Exponential Kernel function evaluated at ξ .

For the mean function written in the form shown in Equation 4.2

$$\mu(t) = \sum_k w_k k(t, t_k) \quad 4.2$$

where w_k is defined in Equation 4.3

$$w_k = \sum_j a_{kj} y_j \quad 4.3$$

where a_{kj} is an element in matrix \mathbf{A} defined in Equation 4.4,

$$\mathbf{A} = (\mathbf{K}_{X,X} + \sigma_y^2 \mathbf{I})^{-1} \quad 4.4$$

its Fourier transform is shown to be Equation 4.5.

$$\mathcal{F}[\mu](\xi) = \mathcal{K}(\xi, 0) \sum_k w_k e^{-i\xi t_k} \quad 4.5$$

4.2 Finding the Posterior Covariance of the Fourier Transform

The same steps can be used to derive the equation for the Fourier transform of the function for the main diagonal of the Covariance Matrix when written in the form shown in Equation 4.6

$$\Sigma(t) = k(t, t) - \sum_k v_k k(t, t_k) \quad 4.6$$

where v_k is defined in Equation 4.7

$$v_k = \sum_j a_{kj} k(t, t_j) \quad 4.7$$

where t_j is the j^{th} element in \mathbf{X} .

The resulting Fourier transform can be written as shown in Equation 4.8

$$\mathcal{F}[\Sigma](\xi, \xi) = \mathcal{K}(\xi, 0)\delta(\xi - \xi) - \sum_k \sum_j a_{kj} e^{-i\xi(t_j+t_k)} \mathcal{K}(\xi, 0)^2 \quad 4.8$$

where δ is the Dirac delta function, which ensures that the two inputs are equal.

Only the main diagonal was computed because this is all that is required to plot the standard deviation of the output predictions (which is the square root of the covariance), and because hyperparameter training (which would require the full matrix) was completed in the time domain already. This decreased the compute requirements significantly, from $O(N^2)$ to $O(N)$, where N is the number of frequency predictions required.

5 Results and Discussion

After this Gaussian Process Model was complete, it could then be set up and used in a variety of test cases to explore its advantages and disadvantages versus the Fast Fourier Transform. The dataset used the same model as in Figure 3.1, but the number of datapoints was extended to 1024 (at the same sample rate of 32 Hz) to provide more information, allowing for more accurate results.

5.1 The Basic Dataset

5.1.1 Accuracy of the Fourier Transform

Initially, no noise was added in order to verify the new approach of computing the Fourier transform. Since the dataset was simulated using known parameters, the exact Fourier transform of the latent function could be calculated analytically. This could be used to test the accuracy of the GP model, using the performance of the FFT (with a Hanning window) on the same system as a benchmark. Here, the posterior mean of the GP's Fourier transform, as well as the FFT and the analytical Fourier transform were calculated. Since the Fourier transform is complex, to show it fully in graph form it must be broken down into two components - either into its magnitude and phase, or into its real and imaginary parts. Both representations are shown in Figure 5.1 for each of the three variants of the Fourier transform.

To verify the model, the properties of the peak were compared to the exact simulated value for each. From observation of the magnitude plots, the GP Fourier transform seems to more smoothly reach the peak, which resembles the shape of the analytical Fourier transform more closely than

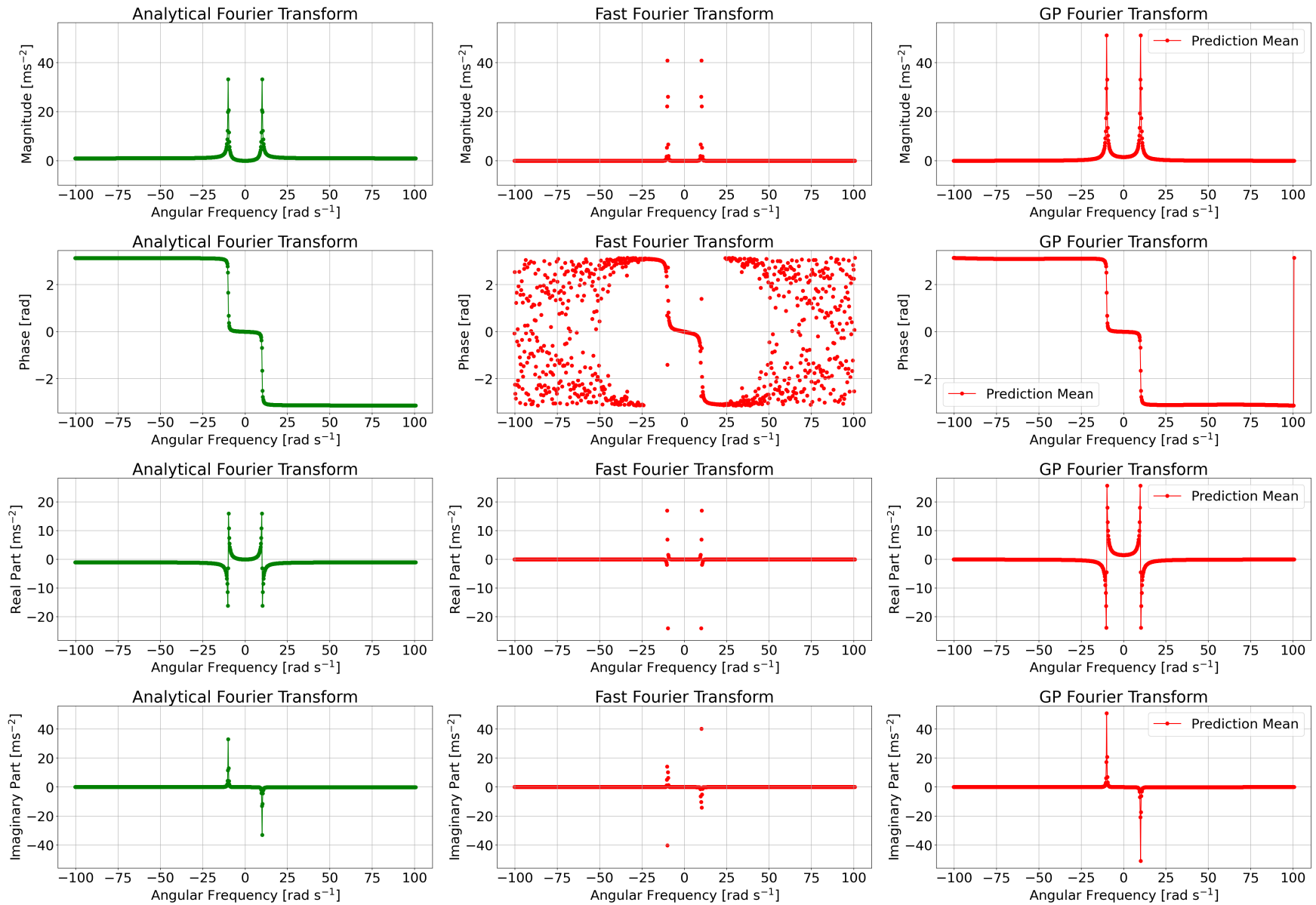


Figure 5.1 - The analytical Fourier transform, FFT and the posterior mean of the GP's Fourier transform for the basic dataset of the SDOF system without noise, broken down into the magnitude and phase components, as well as the real and imaginary components.

the FFT. This suggests that it captures information relating to damping better, likely because the FFT's Hanning window would distort the signal's exponential decay in the time domain from the damping dissipating energy over time. Since the mean of the GP was sampled in the frequency domain in the same places as the FFT, the angular frequency of the peak - as measured by the sample with the largest magnitude - was identical, 10.01 rad s^{-1} , within one sample interval of the analytical peak of 10.00 rad s^{-1} . The percentage errors in the magnitude of the peaks were 22.7 % and 53.5 % for the FFT and GP model respectively.

Next, the Mean Squared Error (MSE) was used to compare the accuracy (lower is better) of each of the FFT and the posterior mean of the GP's Fourier transform with respect to the analytically exact Fourier transform. This algorithm finds the average of the squares of the differences between the points at each index position of two series, as shown in Equation 5.1.

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (a_i - b_i)^2 \quad 5.1$$

where a_i, b_i are the values of vectors **A**, **B** at index i .

The MSE for the magnitude component, as well as the real and imaginary parts of each of the two numerical methods is shown in Table 5.1, where the error of the magnitude GP model's Fourier transform was 1.22 times larger than that of the FFT, but error real and imaginary parts were 4.42 and 12.66 times smaller respectively.

Basic Dataset	FFT	GP FT Mean
Magnitude MSE [ms^{-2}]	2.09	2.55
Real Part MSE [ms^{-2}]	7.59	1.72
Imag Part MSE [ms^{-2}]	10.57	0.83

Table 5.1 - A table showing the Mean Squared Error (MSE) for the components of both the FFT, and the posterior mean of the GP's Fourier transform when compared to the analytically exact Fourier transform for the basic dataset.

Note that the MSE of the phase was not computed because numerical instabilities had a large effect in its values when the real and imaginary components were near zero, so it wouldn't have been meaningful. But overall, the performance of the GP model was comparable to the FFT, since neither was superior in every metric. Therefore, the GP model was shown to be a valid alternative to the FFT.

5.2 Robustness to Noise

The robustness to noise was tested (Objective 3) by calculating the MSE of the two methods when zero mean Gaussian noise with a standard deviation of 25 ms^{-2} was added to the signal. The components of the Fourier transforms are shown in Figure 5.2.

Table 5.2 shows that the GP model performed significantly better here, with the error of the magnitude, and real and imaginary parts of the GP model's Fourier transform being 9.66, 6.48 and 11.64 times smaller than the FFT respectively.

Noisy Dataset	FFT	GP FT Mean
Magnitude MSE [ms^{-2}]	22.80	2.36
Real Part MSE [ms^{-2}]	22.88	3.53
Imag Part MSE [ms^{-2}]	27.94	2.40

Table 5.2 - A table showing the Mean Squared Error (MSE) for the components of both the FFT, and the posterior mean of the GP's Fourier transform when compared to the analytically exact Fourier transform when Gaussian noise with a standard deviation of 25 ms^{-2} was added to the signal.

Furthermore, the magnitude of the peak was found more accurately by the GP model in this case, with a percentage error of 15.8 % versus 38.1 % for the FFT, with no change in the angular frequency of the peak for either method.

Since real world data always contains noise, this Fourier transform that is more robust to noise would have widespread benefits in mechanical engineering because it would allow for more accurate Frequency Response Functions (FRFs) of structures, and therefore more accurate identification of system properties. After the FRF of a structure such as a wind turbine tower is determined, it can be evaluated with the same forces that the system is expected to experience to predict the maximum amplitude of vibration expected during its lifetime. If this is excessive, the system is at risk of mechanical failure or high cycle fatigue, so the design must be changed to prevent this. For example, the mass could be changed to move the resonant frequency (indicated by the angular frequency of a peak) away from the expected forcing frequency, reducing the expected amplitude of oscillation. Alternately, damping could be added to smooth out the resonance peak, reducing its amplitude. Therefore, being able to more accurately determine the FRF and the properties of its peaks through this new GP method would be beneficial in the design process because one would be able to more accurately determine how the design needs to change to reduce the risk of failure to acceptable levels.

5.3 Non-Uniform Sampling of Frequency Domain

Since the Fourier transform is a continuous function, it can be sampled at any set of points in the domain $(-\infty, \infty)$. So, taking the basic dataset and clustering an additional 100 samples of the

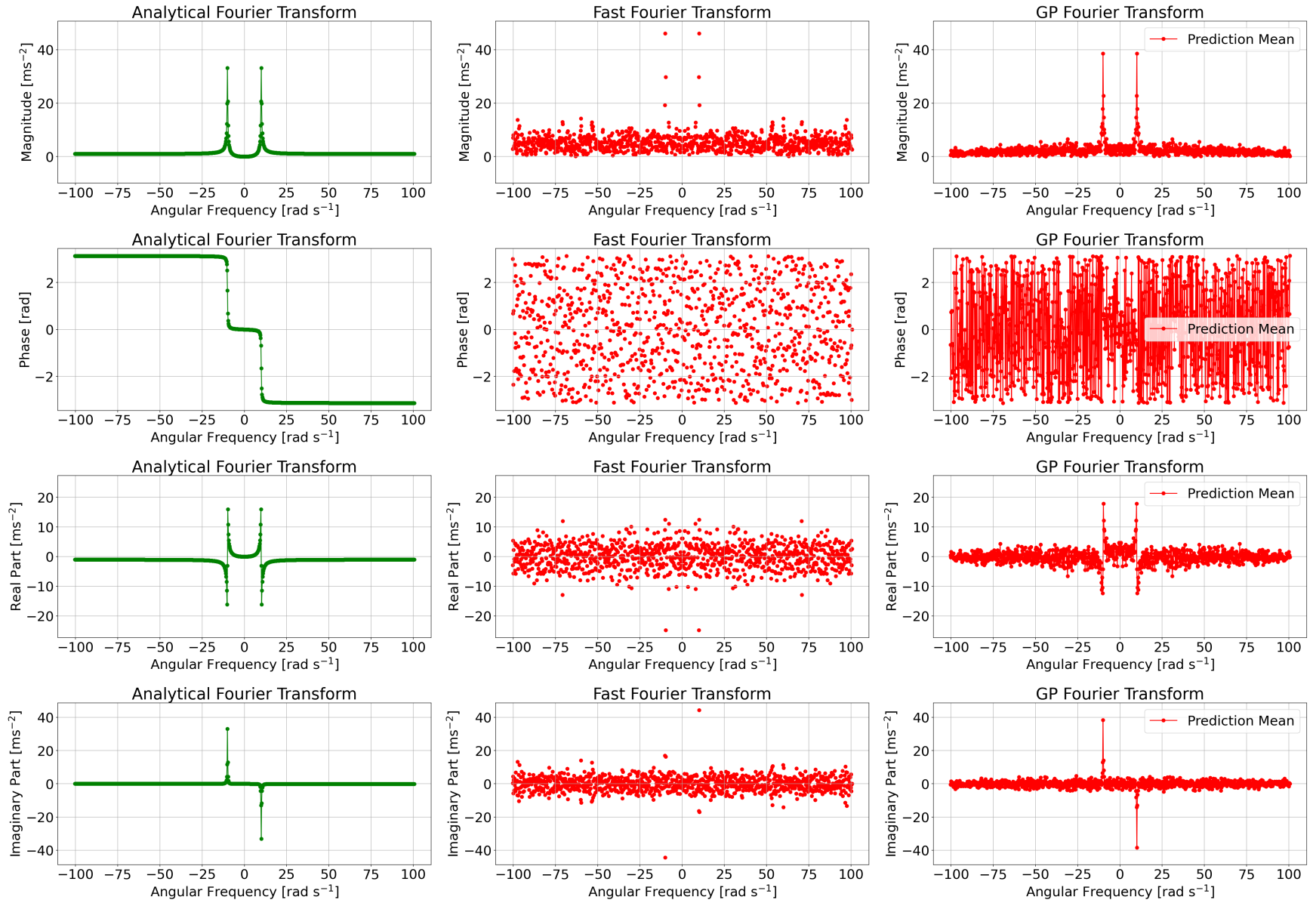


Figure 5.2 - The analytical Fourier transform, FFT and the the posterior mean of the GP's Fourier transform when Gaussian noise with a standard deviation of 25 ms^{-2} was added to the signal, broken down into the magnitude and phase components, as well as the real and imaginary components.

spectrum of the GP in the interval of 1 rad s^{-1} containing the peak was able to improve the error in the calculation of the angular frequency of the peak from the same as the FFT ($0.0149 \text{ rad s}^{-1}$) down to $0.0039 \text{ rad s}^{-1}$. However, the error in the magnitude of the peak (54.2 %) was still larger than that of the FFT (22.68 %).

If the time domain GP was an exact fit to the latent function, as the interval of angular frequency between samples around the peak tended to zero, so would this error. This is not possible with the FFT because the location of the spectrum samples are determined by the time domain samples, both of which must be evenly spaced, meaning that spectrum samples could fall either side of a sharp peak, which creates an uncertainty of \pm half the sample interval.

Furthermore, since the frequency interval is inversely proportional to the total time period, the difference between the error of the peak of the GP model and the FFT is more pronounced for data over a shorter time period. For instance, using only the first 64 points of the same dataset gave the FFT an error in the angular frequency of the peak of 0.574 rad s^{-1} versus 0.054 rad s^{-1} for the GP model - a significant improvement. In addition, the error in the magnitude of the peak (60.64 %) even beat that of the FFT (685 %), which performed very poorly for this short dataset. This means that this proposed GP model may be better suited for use cases in which there is limited time for testing, such as in the aforementioned example of hypersonic wind tunnel tests of scramjets.

5.4 Non-Uniform Sampling of the Time Domain and Overcoming Aliasing and the Nyquist Limit

Similarly, the time domain input data need not be uniformly spaced in order to fit the GP. This would allow the model to be used to find the Fourier transform of datasets with missing data, which may occur, for example, if an accelerometer placed on a gas turbine engine during a test failed to transmit data for a period of time due to a loose connection. If enough data was still present to faithfully represent the system, the GP would not be significantly affected which would allow the test data to still be used. On the other hand, if the standard FFT was used, the test would either need to be repeated at added cost, or additional steps would need to be taken to preprocess the data, adding complication.

However, more impressively, this feature allows frequencies to be detected that would be above the Nyquist limit for the FFT with the same number of datapoints over the same sampling period. This is because for a given time period and number of samples available, many can be bunched in a portion of the time period in order to capture the behaviour of high frequency signal components, leaving a few points over the full period of time to capture well damping information related to the exponential decay of the signal. For example, when increasing the model stiffness to $k = 20,000 \text{ Nm}^{-1}$, and fitting the GP to datapoints sampled over the period at random points in a triangular distribution, it was able to detect the true peak at 141.4 rad s^{-1} with an error of less than

0.1 rad s^{-1} . The FFT of an equivalent dataset containing the same number of datapoints sampled evenly over the same time period for the same system peaked at 59.7 rad s^{-1} , which is an artifact of aliasing caused by the true peak being reflected about the Nyquist limit of 100.5 rad s^{-1} .

This feature of the GP model could be used in contexts where the identification of high frequencies as well as damping information is required, yet there are tight constraints on memory such as in the aerospace industry. Here, a microsatellite must monitor the heavy vibration it experiences on ascent to orbit to ensure the components onboard haven't exceeded their structural limits, which could indicate damage. This would likely include Fourier analysis of accelerometer data to monitor vibration in the frequency domain. However, the memory available to do this is limited because storage capacity has weight, every gram of which significantly increases the fuel requirements and cost to get the satellite to orbit. Therefore, this ability to detect higher frequencies than an equivalent FFT using the same amount of data and memory would be valuable here.

5.5 The Posterior Covariance of the Fourier Transform

The spectrum prediction of the GP model at a given angular frequency is a Gaussian distribution, with a standard deviation around the mean derived from the posterior covariance of the GP's Fourier Transform as shown in Figure 5.3. Here, the standard deviation is largest around 0 rad s^{-1} , and tapers off for higher frequencies. This makes intuitive sense because there are fewer cycles of the lower frequency components in a given time period, so there is less information available to use to detect them.

In the automotive industry, a piston connecting ("con") rod in an internal combustion engine undergoes heavy vibration of very many cycles (twice for each revolution of a four-stroke engine), so fatigue analysis is crucial to ensure that it does not fail from crack formation and propagation before the end of the design life of the vehicle. Since they are typically made of a ferrous alloy such as forged steel, they have a fatigue limit [13], which is a distinct stress level for a given frequency below which failure from fatigue will not occur - even from an infinite number of loading cycles [14]. This stress level can be calculated from experimental acceleration data, and therefore it can be shown that the stress on the con rod is below the fatigue limit by showing that the magnitude of acceleration across the range of frequencies is below its corresponding limit. Testing this would be valuable because one could show that the con rod will not fail from fatigue if the test passes.

However, if the FFT was used, even if the nominal value calculated was below its limit, the uncertainty interval would be unknown and so it would still be unclear whether the upper bound of the range of possible values is above the fatigue limit. Therefore, one could not confidently conclude that the system is below the fatigue limit without a conservative factor of safety.

In contrast, "rather than relying on large factors of safety to account for uncertainty" [15] which

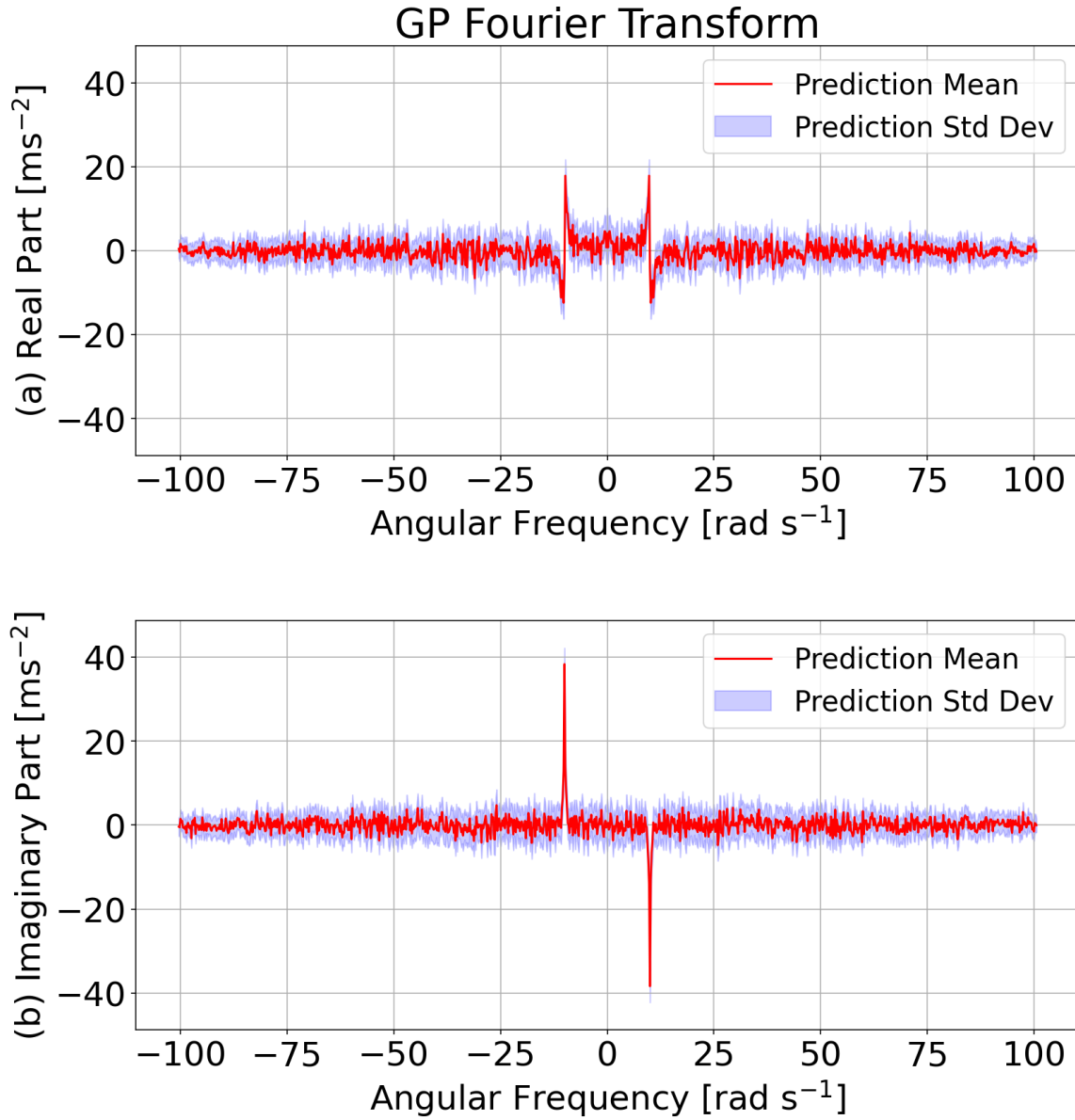


Figure 5.3 - The real (a) and imaginary (b) components of the frequency domain prediction function of the GP including the standard deviation for the noisy dataset.

could add unnecessary weight to overly strengthen the component, the uncertainty information is encoded in the posterior covariance of the GP's Fourier transform. Therefore, it would be possible to calculate a confidence interval of, say, 3σ for the prediction, and if this is entirely below the fatigue limit, one can conclude that the component passes the test without the need for a conservative safety factor, increasing the performance of the engine.

6 Directions for Future Work

6.1 Alternative Kernels

Looking at Figure 3.2, the posterior mean appears to overfit the training data since it jumps from training point to training point, including the noise instead of only fitting to the underlying latent function that varies smoothly. If a different kernel was used that was better suited to the expected shape of the data, the posterior mean may fit the latent function better. For instance, a periodic kernel would be well suited since the latent function is cyclic in nature. To also capture the exponential decay over time, a “compound” kernel could be formed by multiplying it with the Squared Exponential Kernel [11]. Although the posterior mean of the GP’s Fourier transform showed good robustness to noise versus the FFT, if this compound kernel eliminated overfitting, this could significantly improve the robustness to noise of the posterior mean of the Fourier transform further. However, if the kernel is too complicated, the required analytical Fourier transform of the kernel may become intractable.

6.2 Computation Time and Scaling to Large Data

Although this new Gaussian Process (GP) method of performing the Fourier transform is promising, one major disadvantage of this GP model is it struggles to scale large datasets due to the long computation time, especially the hyperparameter training phase since it is of $O(N^3)$, which is poor compared to $O(N \log N)$ for the Fast Fourier Transform (FFT) [9]. This means that the model handles poorly datasets consisting of more than a few thousand points, and even at this scale, fitting the GP to the time domain then converting this to the frequency domain takes several orders of magnitude longer (0.84 s) than the time the FFT (7.5×10^{-5} s) takes to run. This performance is poor considering that it is not uncommon to need to work with datasets consisting of up to a million datapoints in industry, such as during a tap test for modal analysis of a turbine blade, for which the GP model in its current state would be impractical due to the excessive time and cost computational resources that would be required.

To address this, a sparse GP approximation could be employed to allow both the NLML and prediction functions to scale to large data. One type of sparse GP approximation is the Fully Independent Training Conditional (FITC) [16]. This would reduce the complexity of the training phase from $O(N^3)$ to $O(MN^2)$, where M is a set of inducing points chosen to be a subset of the original training points, and $N \gg M$. After this is complete, the hyperparameters could be optimised and the GP could be fit for functions containing more than 10,000 data points in a significantly reduced computation time. However, in practice an implementation could not be found that was numerically stable enough for the NLML values to match the basic GP when the set of inducing points is exactly the same as the test dataset, which they should. This led

to poor convergence and was likely due to the matrix operations involved such as determinants and inverses that can be unstable for large matrices.

To progress further, one may have better performance implementing an alternative sparse GP approximation such as the Variational Free Energy (VFE) approach, which works by approximating the posterior distribution, rather than the prior [9].

7 Conclusion

Overall, this paper explored a new paradigm of “probabilistic numerics” over traditional numerical methods that allowed the Fourier transform to be performed through the use of a Gaussian Process (GP) machine learning model. This model showed significantly better robustness to noise compared to the industry standard Fast Fourier Transform (FFT), exhibiting an improved Mean Squared Error in magnitude versus the true analytical Fourier transform by a factor of 9.66.

The identification of dynamic systems often uses the “Frequency Response Function”, which relies on the “Fourier transform” to convert data from a function of time to a function of frequency. Therefore, the model could allow dynamic systems to be identified more accurately in across industry where data is noisy, from aviation to renewable power generation. This would help to design structures that last longer before failing from fatigue, and that are less likely to fail structurally from excessive excitation under external forces.

Furthermore, the model had several promising features that made it more versatile in this context. Most notably, the fact that the time domain input data need not be evenly spaced enabled it to detect frequencies that would be above the Nyquist limit of an equivalent FFT of the same number of datapoints sampled evenly across the same time period. This makes the model most suitable in industrial contexts where there are constraints to the amount of data that can be collected due to cost, availability, or limited memory.

However, the computational speed of this GP model is currently orders of magnitude slower than the FFT, so before it can be a viable alternative in practice, it would need development to allow it to scale better to large datasets.

References

- [1] Universal Technical Institute. *Motorcycle Suspension Systems Explained: Ultimate Guide*. <https://www.uti.edu/blog/motorcycle/motorcycle-suspension-systems>. Accessed: 2023-12-06. Published: 2020.
- [2] G. Hedau. *The Superposition Principle of Waves | Definition & Examples*. Accessed 25-11-2023. Published: 2023. URL: <https://study.com/learn/lesson/principle-superposition.html>.
- [3] Frederic Grabowski, Paweł Nałęcz-Jawecki, and Tomasz Lipniacki. “Predictive power of non-identifiable models”. In: *Scientific Reports* 13 (2023). Accessed: 10 July 2023, p. 11143. DOI: <https://doi.org/10.1038/s41598-023-37939-8>. URL: <https://www.nature.com/articles/s41598-023-37939-8>.
- [4] *Fourier Transform: Definition, Types, Properties, Applications*. Accessed: 2023-11-23. URL: <https://byjus.com/maths/fourier-transform/>.
- [5] *What are the limitations/shortcomings of Fourier Transform and Fourier Series?* Accessed: 2023-11-23. Published: 2015. URL: <https://math.stackexchange.com/questions/1267754/what-are-the-limitations-shortcomings-of-fourier-transform-and-fourier-series>.
- [6] University of Sheffield. *Advanced Aerospace Propulsion Technology*. University Module. Module code: MEC 463. AUTUMN 2023-24.
- [7] *Probabilistic Numerics*. Accessed: 2023-12-07. URL: <https://www.probabilistic-numeric.org/>.
- [8] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.
- [9] Kevin P. Murphy. *Probabilistic Machine Learning: Advanced Topics*. The MIT Press, 2023.
- [10] Adrian Wills Carl Jidling Niklas Wahlström and Thomas B. Schön. *Linearly constrained Gaussian Processes*. 2017. URL: <https://arxiv.org/pdf/1703.00787.pdf>.
- [11] David Duvenaud. *The Kernel Cookbook: Advice on Covariance functions*. Accessed: 2023-12-06. Published: 2014. URL: <https://www.cs.toronto.edu/~duvenaud/cookbook/>.
- [12] Luca Ambrogioni et al. “The Kernel Mixture Network: A Nonparametric Method for Conditional Density Estimation of Continuous Random Variables”. In: *Proceedings of the Twenty-First International Conference on Artificial Intelligence and Statistics*. Vol. 84. Proceedings of Machine Learning Research. PMLR, 2018, pp. 442–451. URL: <https://proceedings.mlr.press/v84/ambrogioni18a/ambrogioni18a.pdf>.

- [13] RoyMech. *Fatigue Failure and Testing*. Accessed 2012. URL: https://web.archive.org/web/20120415084023/http://www.roytech.co.uk/Useful_Tables/Fatigue/Fatigue.html.
- [14] Ferdinand P. Beer and E. Russell Johnston Jr. *Mechanics of Materials*. 2nd ed. McGraw-Hill, Inc., 1992, p. 51. ISBN: 978-0-07-837340-4.
- [15] Samuel J. Gibson, Timothy J. Rogers, and Elizabeth J. Cross. “Distributions of fatigue damage from data-driven strain prediction using Gaussian process regression”. In: *Structural Health Monitoring* 22.5 (2023), pp. 3065–3076. URL: <https://doi.org/10.1177/14759217221140080>.
- [16] J. Quinonero-Candela and C. Rasmussen. “A Unifying View of Sparse Approximate Gaussian Process Regression”. In: *Journal of Machine Learning Research* 6 (2005).

Appendices

A.1 The Use of Generative AI (ChatGPT)

OpenAI's ChatGPT interface, powered by the GPT-4 large language model, was used to implement the Negative Log Marginal Likelihood (NLML) method in the code in a specific way to ensure numerical stability through the use of the "Cholesky decomposition". The prompt and output is provided in Figure A.1.1.



You

rewrite this line in the most numerically stable way

```
nll = 0.5 * y.T @ np.linalg.inv(K[:, :, i]) @ y + 0.5 * np.log(np.linalg.det(K[:, :, i])) + 0.5 * len(X) * np.log(2 * np.pi)
```



ChatGPT

Computing the inverse and determinant of a matrix can be numerically unstable and computationally expensive. A more numerically stable approach is to use the Cholesky decomposition for the calculation of the log-determinant and to solve the linear system. Here's a refactored version of your line:

```
python Copy code

import numpy as np
import scipy.linalg

# Assuming y, K, and X are defined...

L = scipy.linalg.cholesky(K[:, :, i], lower=True)
alpha = scipy.linalg.cho_solve((L, True), y)

nll = 0.5 * y.T @ alpha + np.sum(np.log(np.diag(L))) + 0.5 * len(X) * np.log(2 * np.pi)
```

Figure A.1.1 - The ChatGPT prompt and output providing a way to compute the NLML that is faster and more numerically stable.

To ensure that this function was valid, its implementation was confirmed by the relevant literature [9] (Section 18.3.6). After which, this function was simplified and adapted so that it could be added to the *gp_model* class as a method, and so that it would work with the way the input data was formatted, as shown in Listing A.1.1.

```

1
2 def compute_nlml(self, hyperparameters):
3     self.update_hyperparameters_and_debug(hyperparameters)
4     self.reshape_X_and_y()
5
6     if self.gp_algo == 'cholesky':
7         K = self.gp_kernel.compute_kernel(self.X, self.X)
8         K += np.repeat(np.array(np.eye(len(self.X)) * 1e-3)[:,: , np.newaxis],
9 self.X.shape[1], axis=2)
10        debug_K = np.squeeze(K)
11        L = scipy.linalg.cholesky(K[: , : , 0], lower=True)
12        n = len(self.y)
13        one_vector = np.ones(n)
14        y_adj = self.y - self.y_mean
15
16        alpha = scipy.linalg.cho_solve((L, True), y_adj)
17
18        term_1_c = (0.5 * y_adj.T @ alpha).item()
19        term_2_c = np.sum(np.log(np.diag(L)))
20        term_3_c = 0.5 * n * np.log(2 * np.pi)
21
22        nlml = term_1_c + term_2_c + term_3_c
23
24        out_c = {
25            'nlml': nlml,
26            'term_1': term_1_c,
27            'term_2': term_2_c,
28            'term_3': term_3_c
29        }
30
31    return out_c

```

Listing A.1.1 - Method used to calculate the Negative Log Marginal Likelihood (NLML).