Non-Parametric Inference of Gravitational Potential of a Galaxy from a Kinematic Snapshot

Yang Hu

Trinity 2023

Contents

1 Introduction

Inferring the gravitational potential of a galaxy is the key to mapping the mass distribution within the galaxy. However, given the precision of current surveys on stellar kinematics, it is impossible to directly measure gravitational acceleration of stars–typically on the order of 1cm $\mathrm{s}^{-1}\mathrm{yr}^{-1}$–which implies the local gradient of gravitational potential. We are only able to get frozen snapshots of stellar kinematics, that is, the stellar phase-space coordinates: positions and velocities. Therefore, it is of pivotal importance in modern astrophysics to infer the gravitational potential of galaxies from limited measurement on stellar phase-space coordinates.

This dissertation addresses the following idealised problem: given precise data $\mathcal{D}$ on position $\boldsymbol{x}$ and velocity $\boldsymbol{v}$ of N stars in a galaxy: $\mathcal{D} = \{\boldsymbol{x}_i, \boldsymbol{v}_i\}, i = 1, 2, ..., N$, where $\mathcal{D}$ is an unbiased sample drawn from an underlying stationary phase-space distribution function (DF) $f(\boldsymbol{x}, \boldsymbol{v})$, how do we infer the gravitational potential $\Phi(\boldsymbol{x})$ of the galaxy?

There have been many previous works on this problem. A vast majority of them constructed analytical models for DF and Φ and performed parameter optimisation [elaborate, cite]. While these parametric approaches can be useful on systems with known form of force law such us the Solar system [cite], they fall short in creating a flexible framework that addresses distributions and potentials with unknown functional forms. In view of the increasingly large sample size of stars with their phase-space coordinate measured by state-of-art galactic surveys such as *Gaia* [cite], rich structures that goes well beyond simple parametric models in phase-space distribution of stellar kinematics in the Milky Way have been identified [cite]. This leads us to consider developing a more flexible model that is applicable to any set of stellar phase-space data and infers the gravitational potential non-parametrically. Another extreme is the attempt via deep learning and neural network [cite], but the interpretability of this approach is not satisfactory.

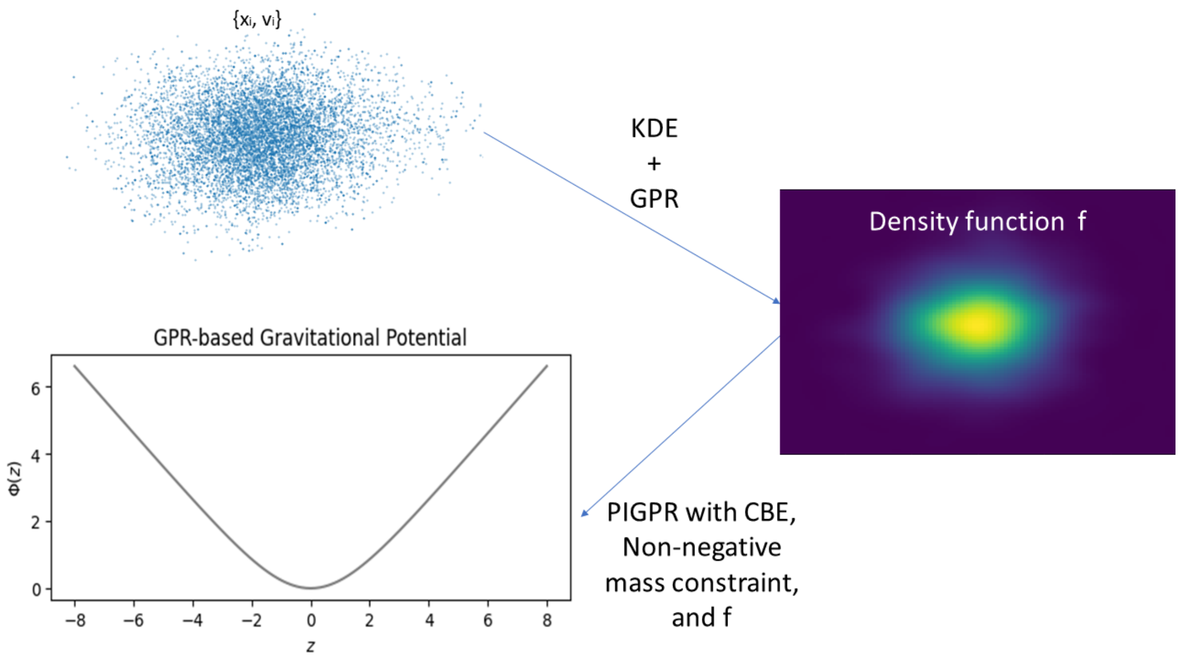
{xi, vi}
KDE + GPR
Density function f
GPR-based Gravitational Potential
Φ(z)
z
PIGPR with CBE, Non-negative mass constraint, and f

Figure 1.1: Overview of the framework

To develop a non-parametric framework and at the same time maintain a high level of interpretability, I borrowed the idea of Gaussian process regression (GPR) from the machine learning community. I used GPR with the training data estimated by kernel density estimation (KDE) to construct a flexible and auto-differentiable model for DF from a given $\mathcal{D}$ and then used physics-informed GPR (PIGPR) to construct a model for Φ from the inferred DF. The link between DF and Φ is collisionless Boltzmann equation (CBE). Figure 1.1 shows an overview of the framework.

The interdisciplinary nature of this dissertation leads me to divide content into various fields in astrophysics and statistics. In chapter 2, I will explain the astrophysical basis and physical assumptions of the model. In chapter 3, I will introduce Bayesian modelling and Gaussian process regression as the core statistical method for inference. In chapter 4 and 5, I will elaborate on applying GPR to stellar phase-space density estimation and CBE solving. In chapter 6, I will apply the framework to simple models of galaxies and show its ability to accurately infer gravitational potential under minimal assumptions. In chapter 7, I will compare this framework with alternative methods and conclude on the work.

2 Collisionless Steady-State Galaxies

Consider a cluster of N stars that forms a galaxy. We define the relaxation time to be the time taken for a star to lose all memory of its initial orbit. Kinetic theory tells us that the relaxation time is given by:

$$t_{\mathrm{relax}} \sim \frac{N}{\log N} t_{\mathrm{dyn}}, \tag{2.1}$$

Where t_{dyn} is the characteristic timescale for a star to collide with another star. Based on current observations, typical galaxies have $N \sim 10^{11}$ stars and are only $\sim 10^2$ dynamical times old. Therefore, the perturbations to a stellar cluster caused by individual stars are largely unimportant. Over timescales $t \ll t_{\mathrm{relax}}$ we can think of stars as test particles moving in a smooth background potential $\Phi(\boldsymbol{x}, t)$ sourced by the distribution of stars and dark matter. This is the collisionless assumption: stars only feel the presence of each other indirectly via the background potential.

We further assume that the galaxy is in steady-state and there is a single, chemically homogeneous population of stars. Then, the dynamics of the galaxy are completely described by a single stationary phase-space distribution function $f(\boldsymbol{x}, \boldsymbol{v})$ and a stationary background potential $\Phi(\boldsymbol{x})$. From 6-dimensional continuity equation, we arrive at the collisionless Boltzmann equation (CBE):

$$\nabla \Phi \cdot \frac{\partial f}{\partial \boldsymbol{v}} - \boldsymbol{v} \cdot \nabla f = \frac{\partial f}{\partial t}, \tag{2.2}$$

where the non-stationarity term $\frac{\partial f}{\partial t}$ should vanish.

Poisson's equation $\nabla^2 \Phi = 4\pi G \rho(\boldsymbol{x})$ tells us the relation between mass density and gravitational potential. So, for the system to be physical, mass density must be non-negative everywhere, and an additional constraint needs to be taken into account:

$$\nabla^2\Phi \geq 0. \tag{2.3}$$

Note that we do not impose that the mass density inferred from DF is consistent with the mass density that generates the gravitational potential. This allows the mass density to contain additional components (such as dark matter) that are not easily measured by some galactic surveys.

CBE provides the crucial direct link between DF and Φ. The idealised problem in chapter 1 thus can be divided into two parts:

1. Given precise data $\mathcal{D}$ on position $\boldsymbol{x}$ and velocity $\boldsymbol{v}$ of N stars in a galaxy: $\mathcal{D} = \{\boldsymbol{x}_i, \boldsymbol{v}_i\}, i = 1, 2, ..., N$, how to model DF where its gradient can be accurately estimated?

2. Given a DF, how to solve CBE for Φ under the constraint $\nabla^2\Phi \geq 0$?.

The first part is a density estimation problem. The second part is solving partial differential equation under constraint. Gaussian process regression (GPR), an increasing popular machine learning method, kills them with one stone. We will demonstrate that, with slight tweaks on conventional GPR, these two problems can be addressed under a flexible Bayesian framework.

3 Gaussian Process Regression (GPR)

3.1 Introduction to Bayesian Modelling

Bayesian Modelling has becoming increasingly popular in recent years in the field of astrophysics, and GPR falls under the Bayesian framework. The foundation that Bayesian modelling lies on is Bayes' theorem. In the context of phase-space density estimation in this dissertation, Bayes' theorem can be phrased as follow. If we observe stellar phase-space coordinate data $\mathcal{D}$ and f is some particular phase-space density function, then the probability that the true phase-space density is this particular f is:

$$P(f|\mathcal{D}) = \frac{P(f)P(\mathcal{D}|f)}{P(\mathcal{D})}. \tag{3.1}$$

$P(f)$ is called prior. It encodes our belief about the DF before we observe data $\mathcal{D}$. $P(\mathcal{D}|f)$ is called likelihood. It is the probability that we observe $\mathcal{D}$ given this particular f. In other words, we update our belief about the true DF based on the data we observe. The posterior $P(f|\mathcal{D})$ is proportional to prior times likelihood, and we want to find the f that maximises the posterior given $\mathcal{D}$.

In the remaining part of dissertation, I will leave italicised font for physical quantities and text font for statistical data. For example, $\boldsymbol{x}$ refers to stellar positions, while $\mathbf{x}$ refers to statistical input data.

3.2 Parametric VS Non-Parametric Approach to Regression

Note that it is not straight-forward how the likelihood $P(\mathcal{D}|f)$ should be found. The approach that I took is to first transform data $\mathcal{D} = \{\boldsymbol{x}_i, \boldsymbol{v}_i\}$ on phase-space coordinates to observations on values of density at some sparse phase-space locations by kernel density estimation. The detail of this transformation will be discussed in section 4.1 when we apply GPR specifically to density estimation problem, but for now let us assume that we obtain this transformed dataset $\mathcal{D}' = (X, \mathbf{y})$ where $X = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_n] = \begin{bmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_n \\ \boldsymbol{v}_1 & \boldsymbol{v}_1 & \cdots & \boldsymbol{v}_n \end{bmatrix}$ contains some phase-space coordinates and $\mathbf{y} = [y_1, y_2, \cdots, y_n]$ contains noise-polluted values of the DF at those phase-space locations. Then finding the likelihood becomes a regression problem: given some points on the DF, what is the entire DF?

There arises two natural ways to tackle a regression problem. One method is to assume a particular form of the DF f parameterised by a list of parameters. The simplest example would be a linear regression model $f = kx + b$ where slope k and y-intercept b are the two parameters we want to optimise. However, this parametric approach restricts the richness of the class of functions we can choose.

An alternative method is, instead of specifying the parameters that determine the form of DF, we only assume the general characteristics of DF and they are controlled by a list of hyperparameters. A non-exhaustive list of hyperparameters include smoothness, ampltiude, lengthscale, differentiability, periodicity, etc. We assign a higher probability to functions with characteristics that match our prior knowledge about f. In other words, we encode our prior knowledge about the function in our choice of hyperparameters. In the end, we would expect to get a distribution in function space rather than parameter space, and this is a non-parametric method for regression.

3.3 Mathematical Foundation of GPR

To do non-parametric regression formally, a powerful and flexible method is Gaussian process regression. A comprehensive guide on GPR can be found in [cite]. Here, we only provide a introductory explanation. A formal definition of Gaussian process (GP) is:

Definition 1. *A Gaussian process is a collection of random variables, any finite number of which have a joint Gaussian distribution.*

In other words, a finite number of random variables {y} satisfy a finite-dimensional multivariate Gaussian distribution that is a slice of an underlying infinite-dimensional Gaussian distribution.

We know that a finite-dimensional Gaussian is completely described by a mean vector and a covariance matrix. Similarly, an infinite-dimensional Gaussian is completely described by a mean function $m(\mathbf{x})$ and a kernel (or covariance function) $k(\mathbf{x}, \mathbf{x}')$. We define $m(\mathbf{x})$ and $k(\mathbf{x}, \mathbf{x}')$ for a real process $f(\mathbf{x})$ as follow:

$$m(\mathbf{x}) = \mathbb{E}[f(\mathbf{x})], \tag{3.2}$$

$$k(\mathbf{x}, \mathbf{x}') = \mathbb{E}[(f(\mathbf{x}) - m(\mathbf{x}))(f(\mathbf{x}') - m(\mathbf{x}'))], \tag{3.3}$$

where $\mathbb{E}$ represents expectation values, and we model the real process as a Gaussian process written as:

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')). \tag{3.4}$$

How does these seemingly abstract definitions tackle a regression problem? To explain this, take our density estimation problem as an example. The real process is the distribution in phase-space of stars (defined in space rather than in time-series). We claim that this DF can be modelled by a Gaussian process and we are free to choose $m(\mathbf{x})$ and $k(\mathbf{x}, \mathbf{x}')$ to specify our belief about the DF before observing any data. The mean function models the 'overall trend' and allow us to incorporate explicit basis functions, while the kernel models the covariance structure between pairs of points on the DF, see section 3.4. Then, from stellar kinematic measurements we have training data $\mathcal{D}' = (X, \mathbf{y})$. We want to know, at a location $\mathbf{x}_* = \begin{bmatrix} \boldsymbol{x}_* \\ \boldsymbol{v}_* \end{bmatrix}$ not in the training input data, what is the DF value f_*?

Note that if we choose a certain kernel k, we can also work out the covariance matrix K of only points we are interested in. For example, for the training input, this is:

$$K(X,X) = \begin{bmatrix} k(\mathbf{x}_1,\mathbf{x}_1) & k(\mathbf{x}_1,\mathbf{x}_2) & \cdots & k(\mathbf{x}_1,\mathbf{x}_n) \\ k(\mathbf{x}_2,\mathbf{x}_1) & k(\mathbf{x}_2,\mathbf{x}_2) & \cdots & k(\mathbf{x}_2,\mathbf{x}_n) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_n,\mathbf{x}_1) & k(\mathbf{x}_n,\mathbf{x}_2) & \cdots & k(\mathbf{x}_n,\mathbf{x}_n) \end{bmatrix}. \tag{3.5}$$

For notatioal simplicity, we assume no prior knowledge on the overall trend and set $m = 0$, although this need not be done, see section 3.4. Given $\mathcal{D}' = (X, \mathbf{y})$, we now want to make prediction on $f(\mathbf{x}_*) = f_*$ at a location $\mathbf{x}_*$ not in the training data. Under the assumption of GP, they should satisfy:

$$\mathbf{y} \sim \mathcal{N}\left(\mathbf{0}, K(X,X) + \sigma_n^2 I\right), \tag{3.6}$$

$$f_* \sim \mathcal{N}\left(\mathbf{0}, K(\mathbf{x}_*,\mathbf{x}_*)\right), \tag{3.7}$$

$$\begin{bmatrix} \mathbf{y} \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) + \sigma_n^2 I & K(X,\mathbf{x}_*) \\ K(\mathbf{x}_*,X) & K(\mathbf{x}_*,\mathbf{x}_*) \end{bmatrix}\right), \tag{3.8}$$

where σ_n is a Gaussian noise added to mimic the uncertainty in ‘measurement’ on DF propagated from stellar kinematic measurement. The detail of this noise will be discussed in section 4.2. We rewrite equation 3.8 as:

$$\begin{bmatrix} \mathbf{y} \\ f_* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} A & C \\ C^T & B \end{bmatrix}\right). \tag{3.9}$$

The conditional distribution of f_* given $\mathbf{y}$ are:

$$f_*|\mathbf{y} \sim \mathcal{N}\left(\mathbf{0} + C^T A^{-1}(\mathbf{y} - \mathbf{0}), B - C^T A^{-1} C\right), \tag{3.10}$$

see, e.g. von Mises [1964, sec. 9.3]. Finally, substitute back the covariance matrices and we get the key result of predictive distribution described by:

$$\text{mean}(f_*) = \bar{f}_* = K(\mathbf{x}_*, X)(K(X, X) + \sigma_n^2 I)^{-1}\mathbf{y}, \tag{3.11}$$

$$\text{Variance}(f_*) = \text{Var}(f_*) = K(\mathbf{x}_*, \mathbf{x}_*) - K(\mathbf{x}_*, X)(K(X, X) + \sigma_n^2 I)^{-1} K(X, \mathbf{x}_*). \tag{3.12}$$

An illustration of a typical GPR on 1-d training data is shown in figure 3.1, where black dots are training data with uncertainty, green dashed line is an attempt to model the data parametrically by a sine function, the solid grey line is the predictive mean function by GPR, the shaded grey zone is the 1-σ confidence interval by GPR. If we choice a kernel k, then everything in equation 3.11 and 3.12 is known, and we can work out $\bar{f}$ at any locations not in the training data with the bonus of knowing the uncertainty in this prediction described by $\text{Var}(f)$. Note that the predictive variance at locations far away from training data is larger than those near the training data, as training data is a strong piece of information that strengthens the certainty on the prediction. This $\bar{f}$ with $\text{Var}(f)$ will be an estimate of the entire f, and hence we conclude the regression problem.

3.4 Mean function and Kernel Selection

We have not specify what mean function m and kernel k we should choose for the specific problem of interest. In the density estimation problem, if we believe that the function space of DF is spanned by some explicit basis functions, then we should set m to those explicit basis functions. In general, if we have no prior knowledge on the 'overall trend', or we believe that as phase-space coordinate $\mathbf{x} \to \pm\infty$, $f \to 0$, we should set $m = \mathbf{0}$ to represent this belief. In solving partial differential equations,

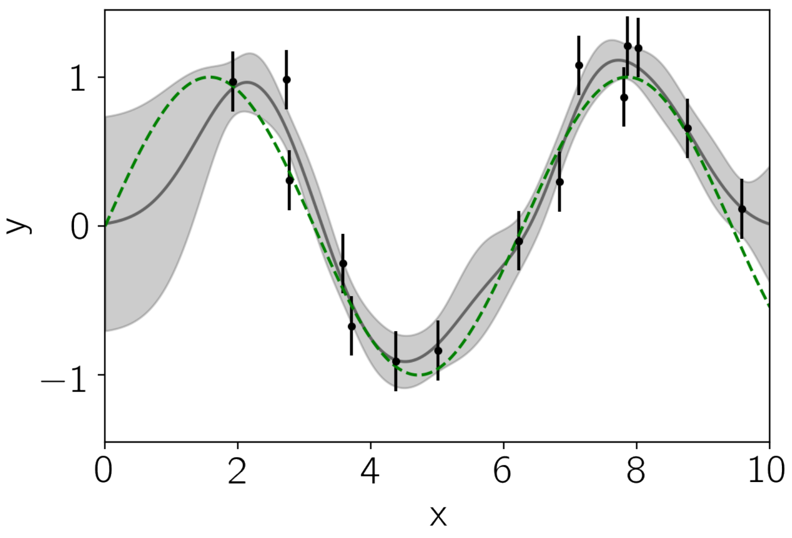

Figure 3.1: An example of GPR predictive mean and variance, figure adapted from george

unless else specified, we also set $m = \mathbf{0}$ to represent our 'ignorance' on the true solution of equation. The remaining question is to understand what kernel really means and how to choose a good kernel.

We said that GPR is under Bayesain framework. If we set the mean function to be $\mathbf{0}$, then the only other place for us to include prior knowledge about the modelled function in this framework is via the kernel. A kernel is an infinite-dimensional generalisation of a covariant matrix that tells the covariance between pairs of points. In our context where measurements appear in space rather than in time-series, consider the following example. We have a galaxy on kpc scale that upon first look, the DF appears to be smooth. Then, it is intuitive to think that two DF measurements that are a few pc away are strongly correlated, while two DF measurements at $\boldsymbol{x}_i$ and $\boldsymbol{x}_j$ that are tens of kpc away have little influence on each other. If we also consider other phase-space dimensions, then this covariance structure can be quantified using stationary kernels that only depend on $|\mathbf{x}_i - \mathbf{x}_j|$.

A popular class of stationary kernels that incorporates our prior knowledge about the differentiabilty (hence smoothness) of the function is Matérn class kernel:

$$k_{M,\nu}(\mathbf{x}_i, \mathbf{x}_j) = \alpha^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu}|\mathbf{x}_i - \mathbf{x}_j|}{l} \right)^{\nu} \mathbb{B}_{\nu} \left(\frac{\sqrt{2\nu}|\mathbf{x}_i - \mathbf{x}_j|}{l} \right), \tag{3.13}$$

where hyperparameter α is the output amplitude, l is the output length scales; Γ is the standard Gamma function, and $\mathbb{B}$ is modified Bessel function of second order. The most important hyperparameter ν controls the degree of differentiability, which in turn affects the smoothness of these functions. Usually, ν is half-integer and its numerator minus one tells us the number of times the function can be differentiated.

The simplest Matérn class kernel has $\nu = \frac{1}{2}$. It represents our prior belief that the function is very rough and can be differentiated zero time:

$$k_{\mathrm{M12}}(\mathbf{x}_i, \mathbf{x}_j) = \alpha^2 \exp\left(\frac{-|\mathbf{x}_i - \mathbf{x}_j|}{l} \right). \tag{3.14}$$

Another extreme of the Matérn class kernel has $\nu \to \infty$, and it is called squared exponential (SE) kernel, or radial basis function (RBF) kernel. It represents our prior belief that the function is smooth (infinitely differentiable):

$$k_{\mathrm{SE}}(\mathbf{x}_i, \mathbf{x}_j) = \alpha^2 \exp\left(\frac{-(\mathbf{x}_i - \mathbf{x}_j)^2}{2l^2} \right), \tag{3.15}$$

Another kernel with in-between differentiability that can be useful for our purpose is the Matérn class 5/2 kernel:

$$k_{\mathrm{M52}}(\mathbf{x}_i, \mathbf{x}_j) = \alpha^2 \left(1 + t_{5/2} + \frac{t_{5/2}^2}{3} \right) \exp\left(-t_{5/2} \right). \tag{3.16}$$

with $t_{5/2} = \sqrt{5}|\mathbf{x}_i - \mathbf{x}_j|/l$.

Finally, we have white noise kernel which incorporates our belief that the covariance

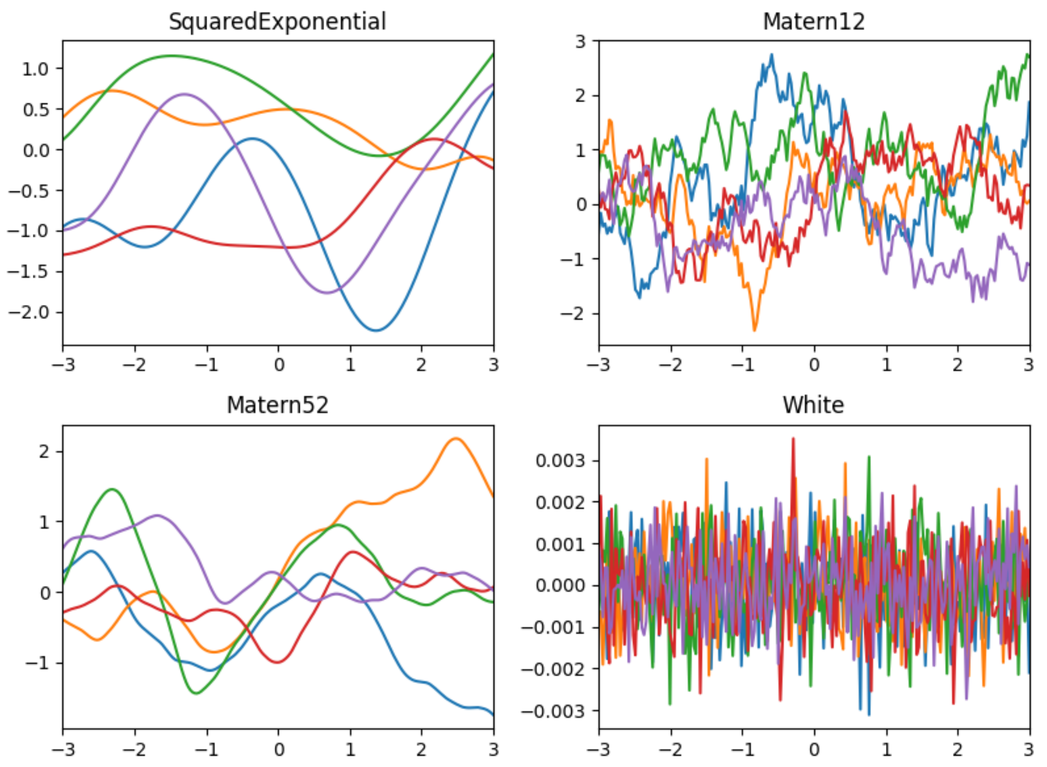

Figure 3.2: Typical shape of Gaussian priors drawn from different kernels

structure is purely white-noise-like:

$$k_{\mathrm{WN}}(\mathbf{x}_i, \mathbf{x}_j) = \sigma_i^2 \delta_{ij}. \tag{3.17}$$

For an illustration of the shape of random Gaussian priors drawn from different kernels, see figure 3.2.

An advantage of kernel is that any linear combination and product of kernels are still valid kernels for GPR [cite]. We can use this property to model complex structures. For example, if we believe that for some exotic galaxy, on top of a smooth background DF, there are some abrupt step-like substructures in phase-space density such as very prominent galactic bars or stellar streams, then our prior belief can be specified by adding k_{M12} to k_{SE} to model an overall smooth DF with global step-like structures. More of this will be discussed in chapter 6 when we apply the framework to specific model of galaxies.

3.5 Hyperparameter Optimisation

After selecting the mean function m and the form of the kernel, we need to set the value of hyperparameters $\mathbf{\Theta} = (\alpha, l, ...)$ in the kernel chosen. If other than the observed data $\mathcal{D}'$ there is no additional constraint, we optimise the hyperparameters by maximising the marginal likelihood of observing output $\mathbf{y}$ given input X and hyperparameters $\mathbf{\Theta}$. The computationally easier way is to minimise the negative of log of this marginal likelihood, or we call it loss function defined as follow:

$$\mathcal{L} = -\ln p(\mathbf{y}|\mathbf{x}, \mathbf{\Theta}) = \frac{1}{2}((\mathbf{y} - m(\mathbf{x}))^T K^{-1}(\mathbf{y} - m(\mathbf{x})) + \ln|K| + n \ln 2\pi), \tag{3.18}$$

where n is the number of training data points and $|K|$ is the determinant of the covariance matrix defined by equation 3.5. Note that this loss function has already included a penalty term $\frac{1}{2}\ln|K|$ that quantifies the complexity of the model so as to prevent over-fitting. If in addition to recovering the observed data and penalty on complexity there are additional constraint on the model, such as preference for symmetry or preference for non-negativity of second derivative, we can add additional penalty terms in the loss function by performing physics-informed Gaussian process regression. The details of modifying loss function will be discussed when we apply the framework to PDE solving in chapter 5.

There are many ways to carry out GPR computationally in Python. Common packages include `george`, `GPflow`, `scikit-learn`, etc. In order to create auto-differentiable model, we use `GPflow` in Python, which further allows efficient gradient calculation on predictive mean with large training dataset to perform Gaussian process regression discussed in this chapter. This concludes the introduction to GPR in this dissertation. Now we need to apply the GPR framework to density estimation problem and partial differential equation solving.

4 Phase-Space Density Estimation

4.1 Density Estimation: Training Data Acquisition

In density estimation problem, in order to apply GPR to get a estimate of the entire DF, one must have the training data for GPR in the first place. However, we only have the training input data X that consists of phase-space coordinate $\mathcal{D} = \{\mathbf{x}\} = \begin{bmatrix} \boldsymbol{x}_1 & \boldsymbol{x}_2 & \cdots & \boldsymbol{x}_N \\ \boldsymbol{v}_1 & \boldsymbol{v}_1 & \cdots & \boldsymbol{v}_N \end{bmatrix}$ but not the training output data $\mathbf{y} = [\mathrm{y}_1, \mathrm{y}_2, \cdots, \mathrm{y}_N]$ that consists of values of DF at those phase-space locations. Getting $\mathbf{y}$ is already a challenge in itself.

There are many ways to obtain an estimate of $\mathbf{y}$. The simplest way is via binning and counting the number of points in the bin to be an estimate of value of DF at the center of that bin. However, the estimated density can be highly dependent on the choice of bin size and location. If the bins are too small, the estimate may be noisy and not capture the true shape of the underlying density. On the other hand, if the bins are too large, important features of the density may be smoothed out or missed entirely. Additionally, binning can be biased if the data is not uniformly distributed across the bins.

A more sophisticated way is kernel density estimation (KDE), the method that I adopted in this dissertation. The idea is to produce a crude estimate of the entire underlying distribution $f(\mathbf{x})$ by adding up multivariate Gaussian distributions of identical shapes and evaluate it at certain locations before passing into the more advanced and flexible machinery of GPR. Mathematically, an estimate of phase-space density f at phase-space coordinate $\mathbf{x}$ by unweighted KDE is:

$$\hat{f}(\mathbf{x}) = \frac{1}{Nh\sqrt{2\pi}} \sum_{i=1}^{N} \exp\left(\frac{-|\mathbf{x}_i - \mathbf{x}|^2}{2h^2}\right), \tag{4.1}$$

where h is the smoothing bandwidth and we have chosen Gaussian Kernels to sum up

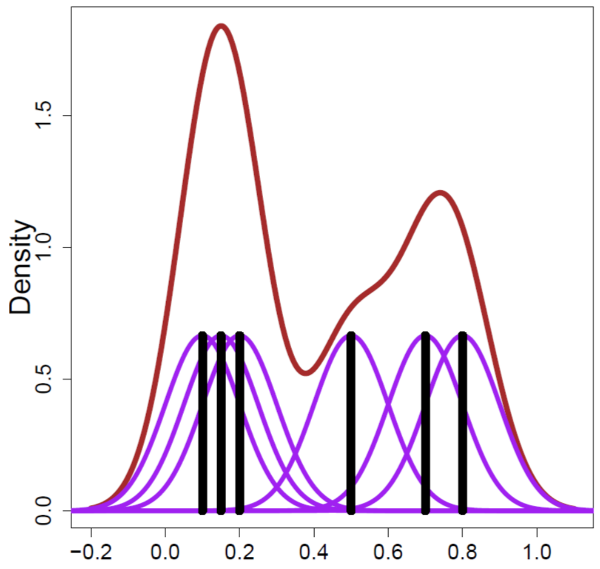

Figure 4.1: In the above picture, there are 6 data points located at where the black vertical segments indicate: 0.1, 0.2, 0.5, 0.7, 0.8, 0.15. The KDE first smooth each data point into a purple density bump and then sum them up to obtain the final density estimate–the brown density curve.

in the absence of any other information about the true f. The selection of the global optimal bandwidth without knowing the true f is an unsolved statistical problem, but there are ways to estimate the optimal h, such as Scott's rule, Silverman's rule and maximum-likelihood cross-validation. For our problem, I found little difference on the final result, so the popular Scott's rule $h = 1.059\sigma N^{-1/(d+4)}$ is adopted, where σ is the standard deviation in space and d is the dimension (for criticism on this rule, see [cite]). Figure 4.1 gives a graphical illustration of KDE.

So, with the optimal h chosen, we can apply KDE to a set of n phase-space points $\{\mathbf{x}\}$. This set of points can be the original set $\mathcal{D}$ used for KDE, or some other set of phase-space locations. I found that for the same number of training input, those evenly spaced out within the domain we want to make prediction produces more accurate estimate of the underlying density. Hence, we will choose to perform KDE on a grid X, and get training output $\mathbf{y}$ at X. In the end, $\mathcal{D}' = (X, \mathbf{y})$ would be the training data to pass onto GPR. Computationally, we use `scipy.gaussian_kde` to perform KDE.

If equation 4.1 gives an estimate of the entire DF, why don't we use it directly as the final estimate of the entire stellar phase-space density function? This is because

KDE alone has only one parameter h that applies to all points and this action of applying identical Gaussian to all points loses the flexibility and fine-tuning in modelling the potentially complex structures of underlying DF. Most importantly, in pure KDE framework there is no place to incorporate our prior knowledge about the DF, and we would not fully use our knowledge about the DF which is not desirable under Bayesian framework.

4.2 Density Estimation: Uncertainty Propagation

Although in our idealised problem we assumed precise measurement of phase-space coordinate, our framework can easily extend to data with measurement uncertainty. For KDE with imperfect data, we incorporate uncertainty in phase-space measurement using weighted kernel function:

$$\hat{f}(\mathbf{x}) = \frac{1}{Nh\sqrt{2\pi}} \sum_{i=1}^{N} w(\mathbf{x}_i) \exp\left(\frac{-|\mathbf{x}_i - \mathbf{x}|^2}{2h^2} \right), \tag{4.2}$$

and the weight is given by:

$$w(\mathbf{x}_i) = \frac{1}{\sigma_\epsilon^2 + \epsilon(\mathbf{x}_i)^2}, \tag{4.3}$$

where $\epsilon(\mathbf{x}_i)$ is the magnitude of uncertainty in $\mathbf{x}_i$ and σ_ϵ is the standard deviation in ϵ.

To propagate the uncertainty from KDE to GPR as the σ_n term in equation 3.6, we need to calculate the uncertainty of $\hat{f}(\mathbf{x}_i), i = 1, 2, ..., n$. Since we don't know the underlying density, a common way is to use k-fold cross-validation to estimate the prediction error and set it to be the uncertainty.

The basic idea is to divide the $\mathcal{D}$ into K subsets or "folds" of roughly equal size. One fold is used as the validation set, while the other $K - 1$ folds are used as the training set. The model is then trained on the $K - 1$ folds and evaluated on the remaining validation fold. This process is repeated K times, with each fold being used once as the validation set. After all K folds have been used as the validation set, we can calculate the mean squared error σ_K for each validation run. We then average these σ_K over all K validation runs to obtain a single estimate σ_n of the model's performance. σ_n^2 would be set as the variance of the Gaussian noise of training data in equation 3.6.

Hence, we conclude the construction of a GPR-based flexible and auto-differentiable DF f with accountable prediction uncertainty. In the following chapter, we will explore how GPR can be applied to go from f to Φ by solving CBE.

5 Solving CBE under Constraint

5.1 Physics-Informed GPR for PDE solving

In section 3.5 we briefly mentioned encoding additional constraints in the GPR framework. This is actually a form of physics-informed Gaussian process regression (PIGPR). PIGPR is a modified GPR that can be used to solve partial differential equations (PDE) by incorporating physical laws and constraints into the model.

Let us see what PIGPR is and how it can be applied to solving PDE numerically under Bayesian framework. The time-independent Collisionless Boltzmann equation (equation 2.2) can be rewritten as:

$$\frac{\partial\Phi}{\partial x}\frac{\partial f}{\partial v_x}\bigg|_{(\boldsymbol{x},\boldsymbol{v})}+\frac{\partial\Phi}{\partial y}\frac{\partial f}{\partial v_y}\bigg|_{(\boldsymbol{x},\boldsymbol{v})}+\frac{\partial\Phi}{\partial z}\frac{\partial f}{\partial v_z}\bigg|_{(\boldsymbol{x},\boldsymbol{v})}-v_x\frac{\partial f}{\partial x}\bigg|_{(\boldsymbol{x},\boldsymbol{v})}-v_y\frac{\partial f}{\partial y}\bigg|_{(\boldsymbol{x},\boldsymbol{v})}-v_z\frac{\partial f}{\partial z}\bigg|_{(\boldsymbol{x},\boldsymbol{v})}=0. \tag{5.1}$$

Usually, CBE is consider as a partial differential equation for f. In this dissertation, however, since we would have an auto-differentiable model of DF f by completing the density estimation problem, CBE is only a linear PDE for Φ. For notational simplicity, first consider CBE with stationary DF $f(x,v)$ in one spatial dimension. 1-d time-independent CBE is written as:

$$\frac{\partial\Phi}{\partial x}\frac{\partial f}{\partial v}\bigg|_{(x,v)}-v\frac{\partial f}{\partial x}\bigg|_{(x,v)}=0. \tag{5.2}$$

Under the Bayesian framework, we should encode our prior knowledge about Φ and any observation to evaluate its posterior. In PIGPR, the following information can be regarded as prior knowledge of Φ or observation on Φ:

1. **Belief on function space of solution.** Overall trend and covariance structure. We have already covered this in section 3.4.

2. **Belief on symmetries.** If we believe that Φ exhibits some spatial symmetry, then in addition to the default loss function of equation 3.18, we can add a term to penalise asymmetry. For example, if we believe that $\Phi(x) = \Phi(-x)$, then we can calculate the mean-square-error of Φ and add

$$\mathcal{L}_{\text{asym}} = \lambda_{\text{asym}} \frac{1}{x_2 - x_1} \int_{x_1}^{x_2} (\Phi(x) - \Phi(-x))^2 \mathrm{d}x \tag{5.3}$$

to equation 3.18, where λ_{asym} quantifies the strength of asymmetry penalty and needs to be carefully tuned for specific problem.

3. **Constraint on non-negativity of second derivative**. Equation 2.3 suggests that we need a heavy penalty for negative mass density. We use the rectified linear unit function ($\text{ReLU}(y) = \max(0, y)$) to define the penalty term:

$$\mathcal{L}_{\text{neg}} = \lambda_{\text{neg}} \frac{1}{x_2 - x_1} \int_{x_1}^{x_2} \left|\left|\text{ReLU}(\nabla^2 \Phi)\right|\right|^2 \mathrm{d}x, \tag{5.4}$$

where $||\cdot||^2$ is the standard L_2 norm and λ_{neg} quantifies the strength of negativity penalty.

4. **Observations on boundary condition**. Let Φ be a scalar field defined on domain $\mathbb{D}$ with boundary $\partial\mathbb{D}$ and boundary condition $\Phi|_{\partial\mathbb{D}} = g$. We observe that the boundary condition holds at a finite set of points $X_{\text{BC}} \subset \partial\mathbb{D}$, and $(X_{\text{BC}}, g|_{X_{\text{BC}}})$ will be part of the training data $(X, \mathbf{y})$.

5. **Observations via CBE**. We can condition Φ on the fact that CBE holds at a finite sequence of well-chosen collocation points X_{PDE}. The choice can be based on local minima of equation 3.12 since at these points we know f locally with least uncertainty. If X_{PDE} is dense enough, we obtain a good approximation to the exact conditional process. These observations $(X_{\text{PDE}}, \Phi|_{X_{\text{PDE}}})$ will be the bulk of the training data $(X, \mathbf{y})$.

6.**Any direct observations** on Φ can be included in the training data set. However,

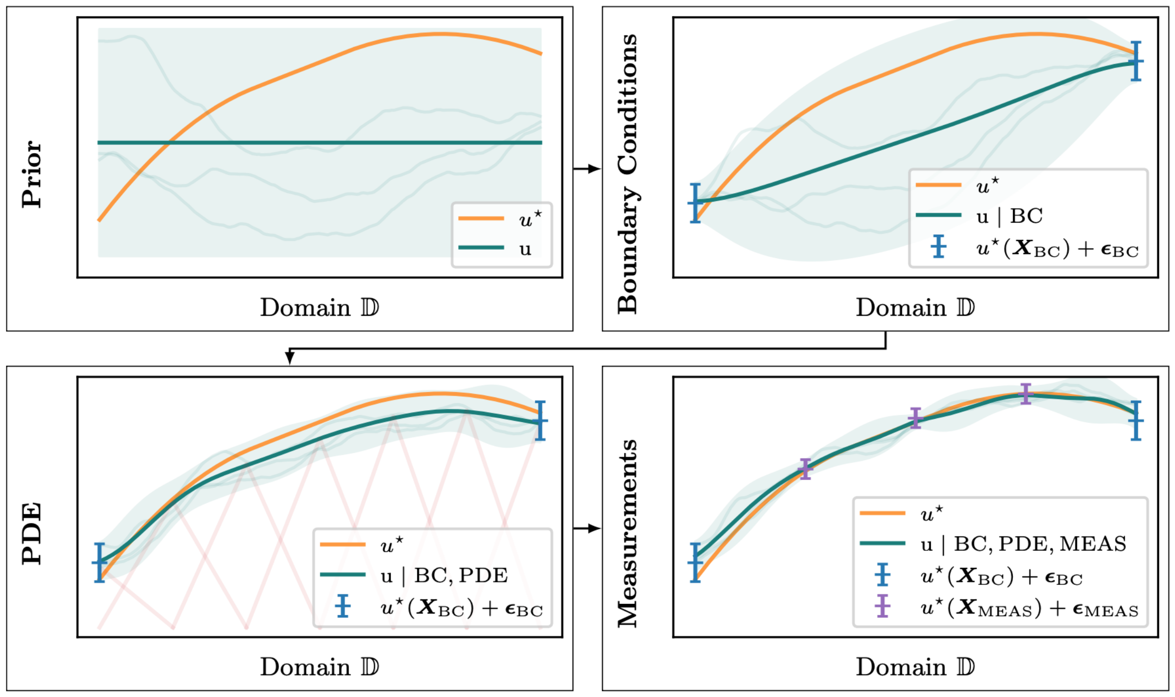

Figure 5.1: Solving a PDE as a learning problem: prior$(1,2,3)\rightarrow$BC$(4)\rightarrow$PDE$(5)\rightarrow$direct measurement(6). u indicates inferred solution and u^* is the true solution. Note however that in PIGPR the 4 steps would be conducted simultaneously, each carrying a weight defined by the loss function.

direct measurement is rather unlikely in our problem.

Figure 5.1 shows the framework described above for PIGPR on PDE solving. The rest of PIGPR progresses as section 3.3. In the end, we should get the predictive distribution for Φ in terms of predictive mean $\bar{\Phi}(\boldsymbol{x})$ and predictive variance $\mathrm{Var}(\Phi(\boldsymbol{x}))$, see equation 3.11 and 3.12.

5.2 Robustness Issue on Solution of CBE

A final remark on solving CBE for Φ without invoking Poisson's equation is that we can use stars with different velocity representations to get $\Phi(\boldsymbol{x})$. In principle, to get $\Phi(\boldsymbol{x})$ from equation 5.1, we only need to evaluate all the partial derivatives of f with a single value of velocity $\boldsymbol{v}$. If we do another evaluation using different $\boldsymbol{v}$, ideally we should get identical $\Phi(\boldsymbol{x})$. However, as our framework estimates f numerically, there will inevitably be fluctuations on solution $\Phi(\boldsymbol{x})$ for different velocity representations. We need to examine the robustness of this framework by checking predicted $\Phi(\boldsymbol{x})$ for different velocity representations.

To obtain a single estimate using all information about $f(\boldsymbol{x}, \boldsymbol{v})$, we can compute the weight of a velocity representation $\boldsymbol{v}$ as the ratio of the generalised volume enclosed by square of 1-σ interval of this representation to the sum of all such volumes for all velocity representations $\{\boldsymbol{v}\}$:

$$w(\boldsymbol{v}) = \frac{\int \mathrm{Var}(\Phi(\boldsymbol{x})|_{\boldsymbol{v}})\mathrm{d}\boldsymbol{x}}{\sum_{\{\boldsymbol{v}\}} \int \mathrm{Var}(\Phi(\boldsymbol{x})|_{\boldsymbol{v}})\mathrm{d}\boldsymbol{x}}, \tag{5.5}$$

and then compute the weighted average predictive distribution for $\Phi(\boldsymbol{x})$:

$$\bar{\Phi}(\boldsymbol{x})_{\mathrm{final}} = \sum_{\{\boldsymbol{v}\}} w(\boldsymbol{v})\bar{\Phi}(\boldsymbol{x})|_{\boldsymbol{v}}, \tag{5.6}$$

$$\mathrm{Var}(\Phi(\boldsymbol{x}))_{\mathrm{final}} = \sum_{\{\boldsymbol{v}\}} w(\boldsymbol{v})\mathrm{Var}(\Phi(\boldsymbol{x}))|_{\boldsymbol{v}}. \tag{5.7}$$

This concludes the application of GPR to solving CBE based on a given f. Together with chapter 2,3 and 4, the complete methodology used in this dissertation has been thoroughly explained. The final predictive distribution of Φ given by equation 5.6 and 5.7 is our inference on the gravitational potential from a kinematical snapshot $\mathcal{D}$. In the remaining part of this dissertation, I will apply this framework to some galactic models and compare it with alternative non-parametric methods.

6 Application to Galactic Models

6.1 Isothermal Disk

We first apply the framework to a self-gravitating isothermal slab. A self-gravitating isothermal slab has only one vertical spatial dimension z and one vertical velocity component v_z, so the DF is 2-dimensional, and it can be used to describe stars close to the galactic disk. We assume that the Hamiltonian is given by $H(z, v) = \frac{1}{2}v^2 + \Phi(z)$. Then, the ideal gravitational potential and DF are given by (see, e.g. [cite])

$$\Phi(z) = 2\sigma^2 \ln\left[\cosh\left(\frac{1}{2}z/z_0\right)\right], \tag{6.1}$$

$$f(z, v) = N\frac{\rho_0}{(2\pi\sigma^2)^{1/2}} \exp\left[-H(z, v)/\sigma^2\right], \tag{6.2}$$

where $N \approx 0.159$ is a normalisation factor. For simplicity, we set $\rho_0 = \sigma^2/8\pi G z_0^2$, $\sigma = 1$, $z_0 = 1$ and $G = 1$. Figure 6.1 (c) shows an ideal DF given by equation 6.2.

We use Metropolis-Hastings algorithm [cite] in Markov chain Monte Carlo method [cite] to sample 10^6 (comparable to e.g. *Gaia*) precise stellar phase-space coordinate from equation 6.2 as the kinematic snapshot $\mathcal{D}$. Note that even with this size, the sample cannot be regarded as a completely unbiased sample. Then, we apply KDE and compute the training data $\mathcal{D}'$ on a 50×35 grid between $z \in [-10, 10]$ and $v \in [-7, 7]$.

Since we know that isothermal disk should have a smooth and symmetric DF, we choose the RBF kernel (equation 3.15) with a white noise kernel (equation 3.17) and perform hyperparameters optimisation with the default loss function (equation 3.18). With these setting in `GPflow`, we run GPR and predict mean DF and variance on a 201×201 grid between $z \in [-8, 8]$ and $v \in [-5, 5]$. Fig. 6.1 (a) shows the predictive mean and fig. 6.1 (b) shows the predictive variance. The absolute error of the density estimation from the ideal DF is also shown in fig. 6.1 (d), where we can see that we

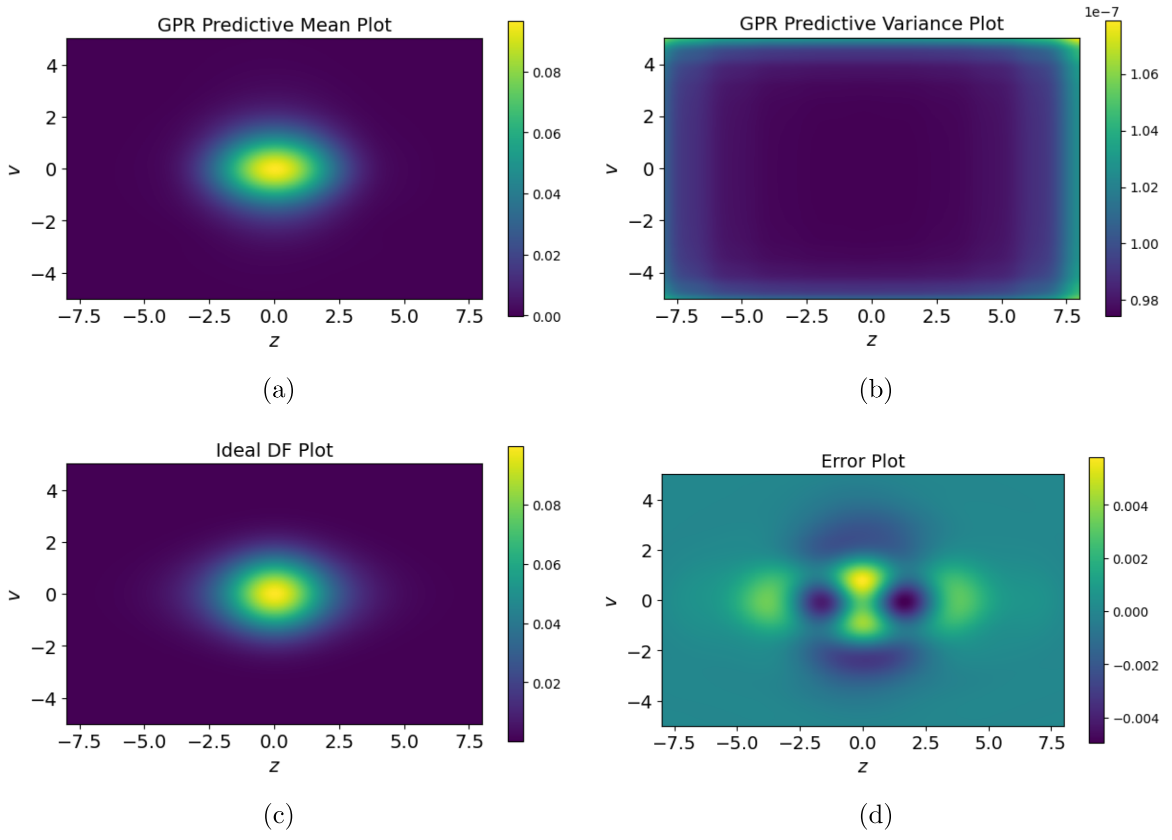

Figure 6.1: Density plots for: (a) GPR predictive mean phase-space DF; (b) GPR predictive variance for phase-space DF; (c) ideal isothermal disk DF; (d) error between GPR predictive mean DF and ideal DF.

accurately recovers the DF within ± 0.006.

The gradient $\frac{\partial f}{\partial z}$ and $\frac{\partial f}{\partial v}$ in both directions at theses grid points are also computed in `GPflow`. From here, we use 1-d CBE (equation 5.2) to compute $\frac{\partial \Phi}{\partial x}$ as the training data and boundary condition for PIGPR. Again, we choose the RBF kernel (equation 3.15) with a white noise kernel (equation 3.17) and perform hyperparameters optimisation with additional negative second derivative penalty ($\lambda_{\text{neg}} = 10$) in the loss function. In the end, we use the weighted estimate of Φ based on variance of each velocity representation (equations 5.6, 5.7) (**note that I have not run on Φ; I just want to have a look on $\frac{\partial \Phi}{\partial z}$ at this stage but going from $\frac{\partial \Phi}{\partial z}$ to Φ up to a constant should be easy.**) to obtain a single estimate on $\frac{\partial \Phi}{\partial z}$. Figure 6.2 shows the inferred $\frac{\partial \Phi}{\partial z}$ i.e. the force law of the isothermal disk. We can see that the inferred force law is most accurate at the center of distribution (within $z \in [-4, 4]$) but goes less accurate at the

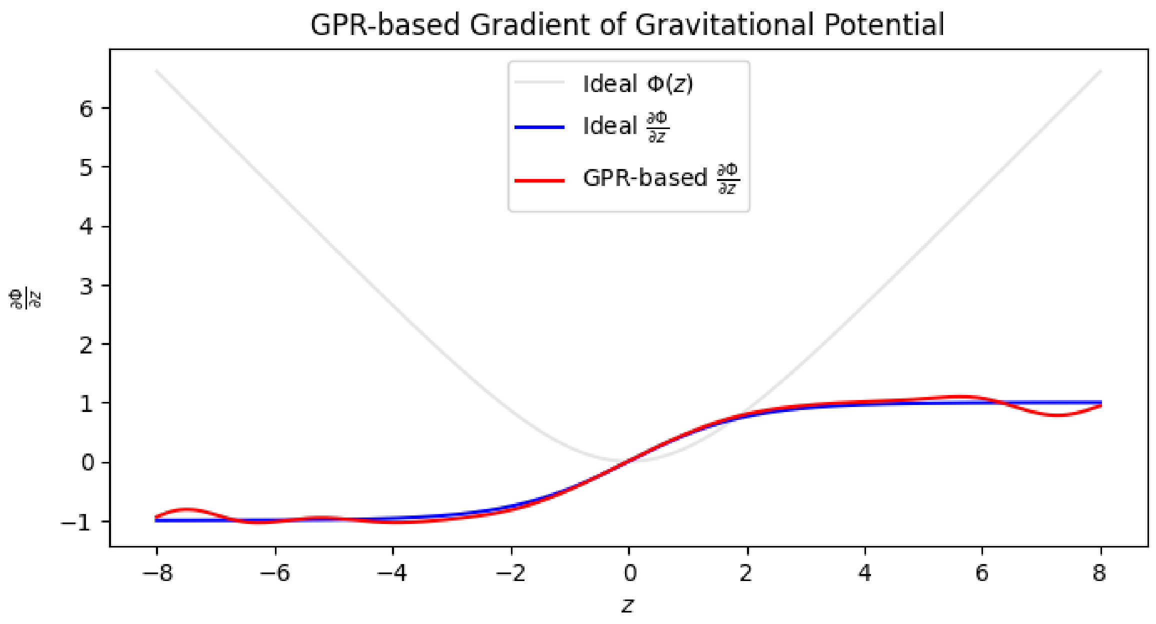

Figure 6.2: Comparison between ideal force law and inferred GPR-based force law for isothermal disk

boundary (an expected property of most statistical method).

6.2 Solar System (To be continued...)

This has not been done but I expect this to be bad because my framework needs large training data to get accurate prediction mean and varaince. An alternative option is to explore other structures, e.g. stellar streams? To be continued...

7 Comparison with Alternative Methods

To be continued...

7.1 Dirichlet Process Mixture Model

Model f prior by Dirichlet Process Mixture. Magorrian (2013)

7.2 Deep Potential

Model f by Normalising flow, then solve CBE by NN. Green et al. (2023)

7.3 Concluding Remarks

Why has this straight-forward method not be attempted before?

Need large number of phase-space coordinate to make accurate prediction, computational efficiency, etc.

To be continued...