Epidemics 29 (2019) 100362

Contents lists available at ScienceDirect

Epidemics

journal homepage: www.elsevier.com/locate/epidemics

Spatial analysis made easy with linear regression and kernels

Philip Milton[a,*], Helen Coupland[a], Emanuele Giorgi[b], Samir Bhatt[a]

[a] MRC Centre for Outbreak Analysis and Modelling, Department of Infectious Disease Epidemiology, Imperial College London, London, UK
[b] CHICAS, Lancaster Medical School, Lancaster University, Lancaster, UK

A R T I C L E I N F O

Keywords:
Regression
Random Fourier features
Kernel methods
Kernel approximation

A B S T R A C T

Kernel methods are a popular technique for extending linear models to handle non-linear spatial problems via a mapping to an implicit, high-dimensional feature space. While kernel methods are computationally cheaper than an explicit feature mapping, they are still subject to cubic cost on the number of points. Given only a few thousand locations, this computational cost rapidly outstrips the currently available computational power. This paper aims to provide an overview of kernel methods from first-principals (with a focus on ridge regression) and progress to a review of random Fourier features (RFF), a method that enables the scaling of kernel methods to big datasets. We show how the RFF method is capable of approximating the full kernel matrix, providing a significant computational speed-up for a negligible cost to accuracy and can be incorporated into many existing spatial methods using only a few lines of code. We give an example of the implementation of RFFs on a simulated spatial data set to illustrate these properties. Lastly, we summarise the main issues with RFFs and highlight some of the advanced techniques aimed at alleviating them. At each stage, the associated R code is provided.

1. Introduction

The mapping of infectious disease has become a cornerstone of global health. Maps can represent the distribution of an infectious disease through space and often time. From pre-intervention planning through to near near-elimination settings and from the global to the village-scale, mapping can inform policy and decision making (Tatem et al., 2010; Noma et al., 2002; Cuadros et al., 2017; Gleason et al., 2017; Mena et al., 2016). Despite their importance, only a small subset of infectious diseases have been comprehensively mapped, estimated at 4% (Hay et al., 2013). As the world becomes increasingly connected, it is likely that more data will become available to enable mapping of many more diseases. Multiple methods have been developed to allow for the flexible and non-linear analysis of spatial data, particularly kernel methods including Gaussian processes. While these models proved excellent flexibility to model the complex dynamics of infectious diseases through time and space, they can be computationally intensive. Combining computationally intensive algorithms with lots of data can require greater resources than what is available to the average researcher. The goal of this paper is to introduce a powerful computationally favourable new approach called random Fourier features (RFF), that represents a simple method to extend kernel methods to large spatial problems.

Various papers on RFF have shown they can provide significant computational speed-ups for kernel methods with minimal loss in accuracy, both theoretically and in practice (Rahimi and Recht, 2007; Yang et al., 2012; Avron et al., 2018). However, these papers are mathematically rigorous and it is not always apparent how these methods work, nor how to incorporate them into spatial analysis. This paper aims to provide the reader with a fundamentals of the RFF method works and illustrate how RFF can be incorporated into existing spatial methods using only a few lines of code. The first half of the paper introduces a large body of theory known as model-based geostatistics (Diggle et al., 1998), building from linear models to kernel methods, specifically kernel ridge regression, capable of capturing complex non-linear spatial patterns. This paper focuses on the modelling of spatial processes but these methods naturally extend to spatio-temporal processes. The second half of the paper focuses on RFF, showing how they can speed up kernel methods by approximating the implicit feature mappings associated with shift-invariant kernels. The paper concludes with a discussion of some of the advantages and disadvantages of RFF and highlights some of the advanced methods to improve upon standard RFF. Code is presented wherever relevant, include a brief toy example in R where we fit a spatial problem using nothing more than linear regression and some transforms.

There is considerable overlap between our introduction of kernel learning and the more traditional formulations based on Gaussian process regression. For an introduction to the Gaussian process and

* Corresponding author.
E-mail addresses: PM5215@ic.ac.uk (P. Milton), hlc17@ic.ac.uk (H. Coupland), e.giorgi@lancaster.ac.uk (E. Giorgi), s.bhatt@ic.ac.uk (S. Bhatt).

https://doi.org/10.1016/j.epidem.2019.100362
Received 22 February 2019; Received in revised form 5 August 2019; Accepted 19 August 2019
Available online 21 August 2019

model-based geostatistics, we refer the reader here (Rasmussen and Williams, 2005; Diggle and Ribeiro, 2007). For a detailed description of the mathematical correspondence between kernels and Gaussian processes, we refer the reader here (Kanagawa et al., 2018).

2. Linear model

In spatial analysis, the aim is to find a model that coverts a set of inputs to a corresponding output of interest at N locations in space and time. The output is termed the *response variable*, and represents the variable that we are trying to predict at each of the N locations, such as case counts or prevalence for a disease under investigation (Gething et al., 2016) (or ancillary epidemiological variables including anthropocentric indicators like height and weight (Osgood-Zimmerman et al., 2018; Josepha et al., 2019) or socioeconomic indicators such as access to water or education (Graetz et al., 2018; Andres et al., 2018)). The inputs are referred to as *explanatory variables* and consist of multiple independent variables recorded at the same locations as the response variables. The choice of explanatory variables is highly dependent on the specifics of a disease, but common examples are population size, age, precipitation, urbanicity and spatial or space-time coordinates. Probably the simplest model to link the explanatory and response variables is the linear model. A linear model assumes that responses are a weighted linear combination of the explanatory variables with some additional uncorrelated (independent) noise, $\boldsymbol{\epsilon}$, which may be written as (McCullagh and Nelder, 1989):

$$\mathbf{y} = X\mathbf{w} + \epsilon \tag{1}$$

where $\mathbf{y} = (y_1, y_2, ..., y_N) \in \mathbb{R}^N$ denotes the vector of response variables given at N locations. The explanatory variables are generally given as matrix $X = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_N] \in \mathbb{R}^{N\times d}$, often referred to as the *design matrix*. The design matrix consists of N rows for each location, and d columns for each explanatory variable. The vector $\mathbf{w} = (w_1, w_2, ..., w_d) \in \mathbb{R}^d$, represents the weights which are used to adjust the influence of each explanatory variable on the model's prediction of the response. For any location i, the model is given as $y_i = \sum_{j=1}^{d} x_{i,j}w_j + \epsilon_i$.

The task is to find an optimal set of weights such the transform of the explanatory variables is as close as possible to the response variables. In order to do this, we first need to define what is meant by “as close as possible” using a *loss function*. A common choice of loss function is the squared loss function that computes the sum of the squared differences between the predicted responses given by the model and the observed response variables for a given set of weights written as:

$$S(\mathbf{w}) = \frac{1}{2}\,||\boldsymbol{y} - X\mathbf{w}||^2 \tag{2}$$

The closer the model's predicted responses ($X\mathbf{w}$) for a given set of weight are to the observed responses ($\boldsymbol{y}$), then the smaller the value of the loss function. Therefore, the set of weights that minimise the loss function will represent the optimal set of weights and give the smallest difference between the model's predicted and the observed response variables. Note, the multiplication by half in Eq. (2) is used solely to simplify the derivative of the loss function and does not affect the solutions. To find the set of weights that minimise the loss function requires solving $\min_{\mathbf{w}\in\mathbb{R}^d} ||\boldsymbol{y} - X\mathbf{w}||^2$. Elementary calculus tells us that this minimal value can be found by taking the derivative of the loss function (with respect to the weights), setting it equal to zero and rearranging the equation to solve for the optimal weights:

$$\begin{aligned} \frac{\partial S(\mathbf{w})}{\partial \mathbf{w}} &= X^T(X\mathbf{w} - \mathbf{y}) = 0 \\ \hat{\mathbf{w}} &= (X^TX)^{-1}X^T\mathbf{y} \end{aligned} \tag{3}$$

The value of the weights that minimises the squared loss function is denoted as $\hat{\mathbf{w}}$, and is called the ordinary least squares (OLS) estimator. Given the explanatory variables at location i, the model's prediction of the response, $\hat{y}_i$, is computed as $\hat{y}_i = \mathbf{x}_i\hat{\mathbf{w}}$.

Minimising the loss function to find weights that best capture the data is referred to as training. The overall model performance after training can be summarised by its error, often calculated as some measure of the difference between the models predicted and the observed response variables. Mean squared error (MSE) is a very common choice which measures the average squared difference between the estimated models predicted and the observed response (MSE $= \frac{1}{N}\sum_i^N (y_i - \hat{y}_i)$). However, a good model should have not only be able to accurately predict the response variables at locations used in training, but also at different locations not used in training the model. To demonstrate the model's capability to do this, a small proportion of the total available data is set aside and excluded from training which forms a test dataset. The error between the predicted and observed response variables at these test locations is calculated to check that the model is able to make accurate predictions at locations that were not used to train the model, this process is termed testing. A model with small training and testing error should be capable of generalising to unobserved location where response variables have not been recorded.

The OLS estimator is a best linear unbiased estimator (BLUE) when the assumptions of the Gauss-Markov theorem are met (see (Davidson et al., 2004) for the standard proof). The Gauss-Markov assumptions include that the error terms, $\boldsymbol{\epsilon}$, have a mean of zero, constant variance and are pairwise uncorrelated. However, there are two critical assumptions required for many analyses. The first is that the response variable is assumed to be a linear function of the explanatory variables specified in the model. Some problems are non-linear such that even the best linear model is a inappropriate representation. The second is that the design matrix, X, is full rank. The design matrix will not be full rank if any of the explanatory variables are perfectly multicollinear, which refers to the situation when one explanatory variable can be expressed as an exact linear combination of one or multiple other explanatory variables (Farrar and Glauber, 1967). When the data contains perfect multicollinearity, it is no longer possible to invert the matrix X^TX, preventing the derivation of the weights in Eq. (3). However, it is more common for variables show multicollinearity (rather than perfect multicollinearity), when there is an approximate but not exact linear relationship between two or more explanatory variables. Multicollinearity is common in epidemiological data where seemingly independent explanatory variables can actually correlate through some latent variable (for example, many variables can be correlated with socioeconomic status) (Vatcheva et al., 2016). In cases of strong (but not perfect) multicollinearity, inversion algorithms may still fail in finding the inverse of X^TX, or generate inaccurate solutions. Except for the assumption of perfect multicollinearity, violating one of more the Gauss-Markov assumptions does not prevent the fitting of a linear model, but results in an estimator that is not the BLUE.

3. A linear model of non-linear features

For many spatial problems the response variables cannot be described as a linear function of the explanatory variables. One approach to introduce non-linearity to the model is to transform the explanatory variables using non-linear transformations, such that the responses are described as a linear combination of non-linear terms. For example, rather than a weighted sum of linear terms (i.e. $x_1 + x_2 + x_3$), we may instead use terms with exponents, logarithms or trigonometric functions (i.e. $exp(x_1) + log(x_2) + sin(x_3)$). Transforming the inputs rather changing the model allows us to continue using the convenient maths we derived for linear models and apply it to nonlinear systems. The transformation of the explanatory variables is called a *feature mapping* and the new space to which the data is mapped is called the *feature space*. Fig. 1 presents an example of a mapping to a feature space such that the response can be expressed in linear terms (code for this example is given in Supplementary Code 1). Generally, a mapping is denoted by Φ and the general form of a feature mapping can be written as:

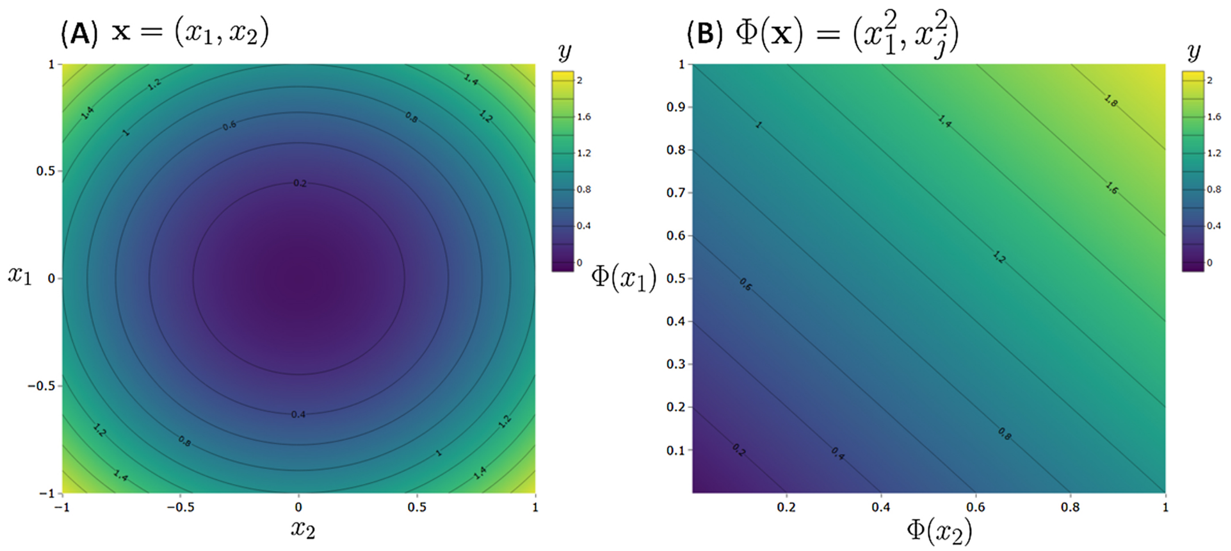

Fig. 1. An example of non-linear feature mapping, where the space is mapped for (A) an input space in which the problem is non-linear into a new feature space (B) in which the outputs can be described as a linear combination of the inputs.

$$\Phi\colon \mathbf{x}_i \in \mathbb{R}^d \mapsto \Phi(\mathbf{x}_i) \in \mathbb{R}^D$$
$$\Phi\colon X \in \mathbb{R}^{N\times d} \mapsto \Phi(X) \in \mathbb{R}^{N\times D} \tag{4}$$

The explanatory variables at location i, $\mathbf{x}_i$, are mapped from a vector of length d to a vector of length D denoted as $\Phi(\mathbf{x}_i)$. Applying this mapping to the entire design matrix gives a new design matrix that exists in the feature space, $\Phi(X) \in \mathbb{R}^{N\times D}$. A mapping can project to into higher-dimensional ($d < D$) or lower-dimensional ($d > D$) space, although in the context of spatial analysis, mapping to a higher-dimensional space is more common. The same set of equations can be used to solve a model that is linear in feature space after the mapping simply by replacing design matrix, X, with the new design matrix in the feature space, $\Phi(X)$:

$$\hat{\mathbf{w}} = (\Phi(X)^T\Phi(X))^{-1}\Phi(X)^T\mathbf{y}$$
$$\hat{y}_i = \Phi(\mathbf{x}_i)\hat{\mathbf{w}} \tag{5}$$

Depending on the choice of mapping, the model is now capable of capturing non-linear relationships between the explanatory and response variables. The solution after the feature mapping now requires solving for the D weights, corresponding to a weight for each column of the design matrix in feature space, $\Phi(X)$. Specifically, the solution is of computational complexity in the best case, where the big O notation, O, is used to denote how the relative running time or space requirements grow as the input size grows. This complexity of is extremely useful because even when the number of locations, N, is large the dimensions of the explanatory variables dominate the solution.

4. Overfitting and ridge regression

Any high-dimensional mapping that significantly expands the number of features ($D \gg d$) greatly increases the risk of overfitting. A model is overfitted if it too closely or exactly predicts the training data, but fails to accurately predict the test data. Overfitting is characterised by a very small training error but a high testing error. All data, but especially epidemiological data, contains random irreducible noise. When a model is overfitted, it captures the random noise in the training data as well as the relationships between the explanatory and response variables. However, this pattern of noise will vary between training and test data. Therefore, while capturing the noise in the training data reduces the training error, it increases testing error by reducing the capability of the model to generalise to the test data which will inherently have different random noise. An inability to accurately predict response variables for test data demonstrates that an overfitted model will not be able to make reliable predictions when presented with data from locations where the response variable has not been recorded. This balance between either reducing the training error or testing error forms the famed result in machine learning termed the bias-variance trade-off (Geman et al., 1992; Domingos, 2000). Overfitting occurs when the model contains more explanatory variables than are appropriate to describe the data, so that the additional variables are fitted to the noise. As an extreme example, if the number of explanatory variables is equal to or greater than the number of locations, then a linear model can pass through every point exactly. Any high dimensional mapping greatly increases the risk of excess explanatory variables and thus overfitting. An approach that is frequently adopted to prevent overfitting is to apply regularisation.

Broadly, regularisation acts to penalise more complex models. Different forms of regularisation define the complexity of a model differently but a common choice of regularisation is called Tikhonov regularisation or ridge regression in the statistical literature (Tikhonov, 1963; Bell et al., 1978; Hoerl and Kennard, 1970). The idea behind ridge regression is to control complexity by penalising models with many large weights. This prevents the model from using the redundant explanatory variables to capture the noise and forces the model to focus on the relationship between explanatory and response. When a model has large weights only a small change in the explanatory variables is required to induce a large change in the response. Therefore, a model with small weights cannot change as quickly as a model with large weights. For a linear model this effectively controls the steepness of the regression line, but for non-linear regression (or linear regression in non-linear feature space) this corresponds to the *wiggliness* of the regression curve. Ridge regularisation consists of two key components; a Euclidean norm term, $||\mathbf{w}||^2$, and and a regularisation parameter λ. The Euclidean norm terms computes the positive square root of the sum of squares of all the weights in the model, $||\mathbf{w}||^2 := \sqrt{w_1^2 + w_2^2 + \cdots + w_d^2}$. A model with many large weights (whether positive or negative) will have a large norm. The $\lambda > 0$ parameter scales the norm and controls the amount of regularisation. Ridge regularisation is achieved by adding a penalisation term to the loss function that depends on the norm of the weights. The loss function for ridge regression is given by:

$$S(\mathbf{w}) = \frac{1}{2}(\mathbf{y} - X\mathbf{w})^2 + \frac{\lambda}{2}\,||\mathbf{w}||^2 \tag{6}$$

When λ is big, the $\lambda\,||\mathbf{w}||^2$ term significantly increases the value of the loss function if the norm of the weights is large so that the act of minimising the loss function favours models with small weights. As λ

approaches zero, large norms are less heavily penalised, allowing for models with larger weights representing optimal solutions to Eq. (6). When $\lambda = 0$ the loss function is equal to the OLS solution. Regularisation results in model weights that are biased towards zero, and thus (by design) the ridge estimate is a biased estimator.

As with the linear model, training requires finding the weights that minimise the ridge regression loss function. This is calculated in the same way, by taking the derivative of the ridge loss function, setting it equal to zero, and solving for the optimal ridge weights $\hat{\mathbf{w}}_{\text{ridge}}$:

$$\begin{aligned}\frac{\partial S(\mathbf{w})}{\partial \mathbf{w}} &= X^T(X\mathbf{w} - \mathbf{y}) + \lambda \mathbf{w} = 0\\ \hat{\mathbf{w}}_{\text{ridge}} &= (X^TX + \lambda I_n)^{-1}X^T\mathbf{y}\end{aligned} \tag{7}$$

where I_n is the identity matrix (a square matrix in which all the elements of the principal diagonal are ones and all other elements are zeros) with $N \times N$ dimensions. The optimal regularisation parameter λ is often unknown but can be estimated during training through methods including cross-validation or restricted maximum likelihood. The solution in Eq. (7) is known as the *primal* solution of the ridge regression and has computational complexity $O(d^2N)$. As with linear regression, the design matrix X can be replaced with $\Phi(X) \in \mathbb{R}^{N\times D}$ to perform ridge regression in feature space with complexity . Note, the addition of λI_n ensures that when $\lambda > 0$ the matrix $(X^TX + \lambda I_n)$ is always invertible even when X is not full rank. Although useful for ill-posed problems and multicollinearity, we consider ridge regression primarily for its ability to prevent overfitting.

5. The dual and the gram matrix

The primal solution to ridge regression required finding the set of d optimal weights, corresponding to a weight for each the column of the design matrix. However, a high dimensional feature mapping ($D \gg d$) significantly increases the number of weights that must be found, with the computational complexity of solving the primal solution exhibiting quadratic growth with the number of features (). An alternative method is called the *dual solution*, that seeks to find optimal variables of length N, corresponding to a variable for each row of the design matrix. Given that the number of rows of the design matrix stays constant when the data is mapped, this dual solution has a computational complexity that is independent of the dimensions of the feature mapping. Consider the following matrix identity:

$$\begin{aligned}(X^TX + \lambda I_n)X^T &= X^T\mathrm{XX}^T + \lambda X^T\\ &= X^T(\mathrm{XX}^T + \lambda I_n)\end{aligned} \tag{8}$$

On the left hand side of the equation, the bracketed terms are multiplied by X^T which may be expanded and then, using matrix algebra, it may be written as an alternative factorisation. By applying this identity, $(X^TX + \lambda I_n)^{-1}X^T$ in Eq. (7) can be replaced with $X^T(\mathrm{XX}^T + \lambda I_n)^{-1}$ to give a new equation for the optimal weights:

$$\hat{\mathbf{w}}_{\text{ridge}} = X^T(\mathrm{XX}^T + \lambda I_n)^{-1}\mathbf{y} \tag{9}$$

This new equation can be further simplified by letting $\boldsymbol{\alpha} = (\mathrm{XX}^T + \lambda I_n)^{-1}\mathbf{y}$. The variables $\boldsymbol{\alpha} \in \mathbb{R}^N$ are the dual variables and, in contrast to the primal solution, correspond to a variable for each row of the design matrix. Substituting $\boldsymbol{\alpha}$ allows us to express the equations for the weights as $\hat{\mathbf{w}}_{\text{ridge}} = X^T\boldsymbol{\alpha} = \sum_{i=1}^{N}\alpha_i\mathbf{x_i}$ which demonstrates that the vector of weights can be written as a weighted linear combination of the N training points. The predicted response given by the model at location i becomes:

$$\hat{y}_i = \mathbf{x}_iX^T\boldsymbol{\alpha} \tag{10}$$

Given that the dual has been derived directly from the primal it is easy to see that both solutions are equivalent, with the two solutions said to exhibit *strong duality* (Boyd and Vandenberghe, 2004). However, the dual form can be derived directly from ridge regression loss function independent of the primal by expressing the problem as a constrained minimisation problem and solving using Lagrangian multipliers (Supplementary Equations 1). The dual solution involves computing and inverting the matrix XX^T, with computational complexity $O(N^3)$ in the worst case. With the dual solution, no matter how high dimensional the feature mapping, the dual solution will only require solving for the same number of dual variables and be of the same computational complexity. Therefore, for any high-dimensional mapping where $D > N$, the dual solution is computationally easier to solve than the primal.

Less obviously, Eq. (9) computes XX^T with the resulting matrix a symmetric positive semidefinite matrix called the Gram matrix, $G \in \mathbb{R}^{N\times N}$. The gram matrix contains the inner products of explanatory variables between every all of N locations. An inner product is a way to multiply vectors together with the result of this multiplication being a scalar measure of the vectors *similarity*. This notion of similarity is central to spatial or temporal analysis where we want to leverage the fact that points close to each other in space or time should be more similar than those far apart. Explicitly, the inner product of two vectors $\mathbf{x_i}$ and $\mathbf{x_j} \in \mathbb{R}^d$ is given by:

$$\begin{aligned}\langle \mathbf{x_i}, \mathbf{x_j}\rangle &= \langle (x_{i,1}, x_{i,2}, \ldots, x_{i,d}), (x_{j,1}, x_{j,2}, \ldots, x_{j,d})\rangle\\ &= x_{i,1}x_{j,1} + x_{i,2}x_{j,2} + \cdots + x_{i,d}x_{j,d}\end{aligned} \tag{11}$$

where $\langle\cdot,\cdot\rangle$ is used to signify the inner product. Therefore, the Gram matrix can be thought of as containing elements which represent the similarity between all pairs of inputs. Taking Eqs. (9) and (10) and substituting XX^T for the full gram matrix, G, and $\mathbf{x}_iX^T$ for the ith row of the gram matrix, G_i (corresponding to the pairwise inner product between location i and all N other locations), lets the dual solution for the weights and model prediction be rewritten as:

$$\begin{aligned}\boldsymbol{\alpha} &= (G + \lambda I_n)^{-1}\mathbf{y}\\ \hat{y}_i &= G_i\boldsymbol{\alpha}\end{aligned} \tag{12}$$

Therefore, the solution to the dual only requires inner products, or in other words, the dual solution only requires similarity between points and not their explicit values. We can exploit this property to allow us to work with the linear regression model in very-high or even infinite-dimensional feature space using kernels.

6. Kernel functions, kernel matrix and the kernel trick

It is rarely apparent *a priori* what the most appropriate mapping is to apply to the data. Therefore, while the dual solution is of constant complexity even in very high dimensional feature spaces, the question remains, which mapping should we use? How many terms should be added? How do we capture interactions between terms? A brute force approach quickly becomes combinatorially large. Given that the solution to the dual is independent of the number of features, and regularisation allows us to limit model complexity to prevent overfitting, the ideal situation would be to define a high-dimensional, if not infinite mapping capable of modelling nearly any function and then apply regularisation. An infinite feature mapping can be expressed as:

$$\begin{aligned}&\Phi\colon \mathbf{x}_i \in \mathbb{R}^d \mapsto \Phi(\mathbf{x}_i) \in \mathbb{R}^\infty\\ &\Phi\colon X \in \mathbb{R}^{N\times d} \mapsto \Phi(X) \in \mathbb{R}^{N\times\infty}\end{aligned} \tag{13}$$

However, we cannot compute the infinite terms required for explicit infinite-dimensional feature mapping. To work with these very high or infinite dimensional spaces, we need to move away from explicit to implicit feature maps which is achieved through kernel functions.

Broadly, kernel functions, denoted as $k(\cdot, \cdot)$, take in the explanatory variables at two locations and directly computes their inner product in a corresponding feature space:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \langle \Phi(\mathbf{x}_i), \Phi(\mathbf{x}_j)\rangle \tag{14}$$

More formally, a kernel function is defined as a two-argument, symmetric positive semi-definite function that corresponds to computing the inner product a corresponding reproducing kernel Hilbert space (RKHS) (Shawe-Taylor et al., 2004). This RKHS and is specific to the chosen kernel function and simply represents a feature space that has a valid inner product (the notion of similarity is conserved). For a detailed explanation of the mathematics of kernels we recommend (Shawe-Taylor et al., 2004). If the kernel function is applied to all pairs of locations in the training data, the resulting matrix generated is termed the *kernel matrix*, $K \in \mathbb{R}^{N \times N}$, and can be thought of as the gram matrix in RKHS:

$$K = \langle \Phi(X), \Phi(X) \rangle = \Phi(X)\Phi(X)^T \tag{15}$$

Importantly, the dual solution only requires inner products to solve. Therefore, the gram matrix in Eq. (12) can be replaced by the kernel matrix:

$$\hat{\mathbf{y}} = K(K + \lambda I_n)^{-1}\mathbf{y} \tag{16}$$

This is equivalent to solving the dual in RKHS associated with the kernel. However, the data is never actually mapped into this implicit feature space. Instead the kernel function direct computes the inner product in the RKHS and uses those inner products to solve the dual. Given that the data is never actually explicitly mapped to RKHS, the feature space can be of very high or even infinite dimensions, as long as we have the associated kernel function, and thus the inner product in this feature space, the the dual can be solved.

The replacement of the explicit mapping with the implicit mapping associated with a kernel function is called the kernel trick and can be applied to many algorithms with dual solutions. Eq. (16) combines ridge regression with the kernel trick to create kernel ridge regression (KRR), that effectively solves the model in the RKHS with the ridge penalty preventing overfitting and ensure small loss for both training and testing. The dual solution still only requires computing the N dual variables with complexity $O(N^3)$. Thus, given an appropriate kernel, the KRR dual can be solved in infinite-dimensional feature space with no added computational complexity. Therefore, the combination of the dual solution and kernels is a powerful tool capable of extending linear models to very high dimensional feature spaces with the ability to handle nearly any non-linear problem.

Kernel functions can be derived by direct construction; finding the function that corresponds to the taking the inner product in the feature space (see Supplementary Equations 2 for an example of constructing the polynomial kernel from an explicit feature mapping). Generally, kernels are described by showing that the proposed function satisfies Mercer's conditions; proving the function is symmetric positive-definite function and guaranteeing there exists a RKHS (Shawe-Taylor et al., 2004). A huge number of kernels are already described, with specific kernels often used to solve specific tasks. Common kernels for spatial analysis include the squared exponential and Matérn kernels. The squared exponential kernel is popular because it generates smooth functions that are appropriate for spatial interpolation, while the Matérn kernel allows a better balance between smoothness and roughness of the resulting functions that may better represent true spatial processes (Stein, 2012). The exponential kernel is popular choices for modelling temporal data, where the similarity between two points is assumed to steadily exponentially decay as the time between them increase. For example, the squared exponential kernel is given by:

$$k_{SE}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{(\mathbf{x}_i - \mathbf{x}_j)^2}{2\ell^2}\right) \tag{17}$$

where $\ell \in \mathbb{R}_{\geq 0}$ is the length-scale. The length-scale controls the smoothness of the resulting functions. The kernel parameters are often learned from the data alongside the model variables. The squared exponential kernel corresponds to the inner product in an infinite dimensional feature space shown in Supplementary Equations 3.

7. The big N problem

One of the primary motivations of combining the dual and kernels is that this process only requires computing and inverting the kernel matrix $K \in \mathbb{R}^{N \times N}$, and solving for the N dual variables, even when the dimensions of the (implicit) feature space is infinitely large. Historically this led to the widespread adoption of kernel methods to solve difficult problems on small/medium datasets. However, the dual still requires $O(N^2)$ storage for N observations and $O(N^3)$ complexity. Given only a few thousand points, these costs can rapidly outstrip the storage and computational power available to most researchers.

A plethora of methods have been developed to allow the scaling of kernels methods to large datasets. These methods aim to find smaller or simpler matrices that provide good approximations of the full kernel matrix. The three main techniques used are low-rank approximations, sparse approximations and spectral methods. Low-rank approximations of a matrix aim to find smaller representations of the kernel matrix that contains all (or nearly all) of the information in the full kernel (Bach and Jordan, 2005). For example, the popular Nyström approximates the full kernel matrix through a subset of its columns and rows (Williams and Seeger, 2001). In contrast, sparse methods aim to find representations of the matrix that are mostly zeros because efficient algorithms exist for the storage of and computation with such matrices (Rue and Held, 2005; Straeter, 1971; Saad and Schultz, 1986). One of the best examples is the sparse matrix generated when modelling spatial data as a Gaussian Markov random field (GMRF) that are solutions to Stochastic Partial Differential Equation (SPDE) (Lindgren et al., 2011; Whittle, 1954, 1963). However, the remainder of this paper will focus on an exciting, new subset of spectral methods called random Fourier features (RFF).

RFF combines the flexibility of kernels with the computational benefits of the primal solution. RFF use the Fourier transform of a kernel function to explicitly map the data to a relatively low-dimensional space that approximates the implicit feature space associated with the kernel. The data in this feature space can either be used to construct an unbiased estimator of the full kernel or be used to solve the primal at a significantly reduced computational cost. This is distinct from many low-rank or sparse approximations that rely on the approximations of kernel matrix and the thus the dual. The RFF method approximates the entire kernel function at once, does not rely on inducing points or "knots", does not require throwing away any data, can easily be incorporated to many existing linear models and has steps no more complicated than sampling from some probability distribution and applying trigonometric functions.

8. Random Fourier features

RFF and other spectral method rely on the characterisation of the kernel function through its Fourier transform. Any function can be decomposed into the periodic, trigonometric functions (sin and cos) of different frequencies that make it up. As an analogy, consider how a complex musical chord is composed of a number of individual notes, where each note is just a string vibrating at a particular frequency. The Fourier transform tells us the how much trigonometric functions of each frequency must be added to construct a given target function. The central idea behind spectral methods is that a good approximation of the frequencies that make up the kernel function will naturally yield a good approximation to the kernel function itself. For the mathematics of Fourier transforms, we refer the reader here (Bracewell and Bracewell, 1986). All spectral methods are based on the same mathematical foundation; the celebrated *Bochner's theorem* (Bochner and Chandrasekharan, 1949). Loosely, Bochner's theorem states that a shift-invariant kernel functions (where the output of the kernel is only dependent on the difference between the inputs and not the explicit values of the input themselves), $k(\mathbf{x}_i, \mathbf{x}_j) = k(\mathbf{x}_i - \mathbf{x}_j)$, on $\mathbb{R}^d$ can be expressed through a Fourier transform (Rudin, 1990):

$$k(\mathbf{x}_i - \mathbf{x}_j) = \int_{\mathbb{R}^d} e^{i\omega^T(\mathbf{x}_i - \mathbf{x}_j)} \mathbb{P}(\omega)\ d\omega \tag{18}$$

This theorem tells us that the Fourier transform of a shift-invariant kernel takes the form of probability distribution, $\mathbb{P}(\omega)$. This distribution is called the spectral density of the kernel and is the distribution of the *amount* of a given frequency, ω, that must be added to construct the kernel function. The larger the spectral density for a given ω, the greater the amount of that frequency must be added to reproduce the kernel function. Applying Euler's identity ($e^{i\pi} = \cos(\pi) + i\sin(\pi)$) to the exponential and ignoring the imaginary component, let us consider Bochner's theorem in terms of the trigonometric functions:

$$k(\mathbf{x}_i - \mathbf{x}_j) = \int_{\mathbb{R}^D} \begin{pmatrix} \cos(\boldsymbol{\omega}^T\mathbf{x}_i) \\ \sin(\boldsymbol{\omega}^T\mathbf{x}_i) \end{pmatrix}^T \begin{pmatrix} \cos(\boldsymbol{\omega}^T\mathbf{x}_j) \\ \sin(\boldsymbol{\omega}^T\mathbf{x}_j) \end{pmatrix} \mathbb{P}(\omega)\ d\omega \tag{19}$$

However, a major problem is that evaluating the integral in Eq. (19) requires integrating over the infinite set of all possible frequencies. To avoid this, we can approximate this infinite integral by a finite one using Monte Carlo integration. In Monte Carlo integration the full integral of a function is approximated by computing the value of the function evaluated at a random set of points and averaging. In RFF, the integral is approximated by averaging the sum of the function evaluated at random samples of $\boldsymbol{\omega}$ drawn from drawn from the probability distribution $\mathbb{P}(\omega)$. The greater the number of samples that are evaluated, the closer the approximation gets to the value of the full integral. Indeed, one of the best properties of random Fourier features is that of uniform convergence of the Monte Carlo approximation of the entire kernel function (rather than pointwise) (Rahimi and Recht, 2007). Therefore, the infinite integral in Eq. (19) can be converted to a finite approximation by taking m independent samples of ω from the power spectral density, and computing the Monte Carlo approximation of the kernel function as:

$$k(\mathbf{x}_i - \mathbf{x}_j) \approx \frac{1}{m}\sum_{s=1}^{m} \begin{pmatrix} \cos(\boldsymbol{\omega}_s^T\mathbf{x}_i) \\ \sin(\boldsymbol{\omega}_s^T\mathbf{x}_i) \end{pmatrix}^T \begin{pmatrix} \cos(\boldsymbol{\omega}_s^T\mathbf{x}_j) \\ \sin(\boldsymbol{\omega}_s^T\mathbf{x}_j) \end{pmatrix}, \quad \{\omega_s\}_{s=1}^{m} \overset{i.i.d.}{\sim} \mathbb{P}(\omega) \tag{20}$$

frequencies from them. For example, generating the frequencies for approximating a squared exponential requires independently sampling frequencies from a Gaussian distribution, or from Cauchy distribution for a Laplacian kernel (Table 1). This is visualised in Fig. 2 that shows different spectral densities (Fig. 2A,C,E,G,I) and the resulting functions produced by sampling from the kernel generated by each spectral density (Fig. 2B,D,F,H,J).

In Fig. 2A, the spectral density is composed of two delta functions such that sampled frequencies, $\boldsymbol{\omega}$, can only take values equal to 1 or 2. The functions generated by sampling from this spectral density show strong periodicity and closely resemble the standard trigonometric functions with corresponding frequencies (Fig. 2B). When the frequencies of the two possible delta functions are increased so that they lie in the set {10, 20}, the functions are again highly cyclical but, due to their higher frequencies, have rougher sample paths and a much smaller period (Fig. 2C,D). By expanding the spectral density to contain five possible frequencies (Fig. 2E,G), the sample paths show considerably more variation due to the inclusion of a larger variety of frequencies (Fig. 2F,H). Finally, Fig. 2I and K show samples functions generated by sampling frequencies form a Gaussian and Cauchy distribution respectively. The Gaussian spectral density corresponds to a spectral density of the squared exponential kernel (Gaussian kernel) and gives rise to smooth sample functions with a huge amount of variety when compared to the simpler spectral densities (Fig. 2J). The Cauchy distribution corresponds to a spectral density generated by the Fourier transform of the Laplacian kernel and generates functions with a high degree of roughness (Fig. 2L) due to the inclusion of very high frequencies in the long tails of the distribution (Fig. 2K). The code for sampling the spectral densities and generating functions is given in Supplementary Code 2.

Eq. (20) shows how the RFF can be used to approximate the kernel function and thus the whole kernel matrix through pairwise application. Incredibly, this can all be written in just 4 lines of R-code:

Code 1 Example of creating random Fourier features to approximate a squared exponential kernel matrix.

```
#X = data matrix of dimensions (N x d), m = number of sampled frequencies
Omega = matrix(rnorm(m*ncol(X)), m) #Squared Exponential kernel
Proj = X %*% t(Omega) #Projection — multiple data with sampled frequencies
Phi = cbind(cos(Proj), sin(Proj)) / sqrt(m) #Fourier feature matrix
K = Phi %*% t(Phi) #Approximation of Kernel matrix
```

When the frequencies are sampled from the power spectral density, the RFF approximation of the kernel function is an unbiased estimator of the kernel function (Rahimi and Recht, 2007). Given that the spectral densities represent probability distributions, it is trivial to sample

Table 1
Common shift-invariant kernels and their spectral densities.

Kernel	Kernel function, $k(\mathbf{x}_i, \mathbf{x}_j)$	Power spectral density, $\mathbb{P}(\omega)$
Squared Exponential [‡]	$\exp\left(-\frac{\lVert \mathbf{x}_i - \mathbf{x}_j \rVert_2^2}{2\ell^2}\right)$	$(2\pi\ell^2)^{-\frac{D}{2}}\exp(-2\pi\ell^2\lVert\omega\rVert_2^2)$
Matén[*], [†], [‡]	$\frac{2^{1-\upsilon}}{\Gamma(\upsilon)}\left(\frac{\sqrt{2\upsilon}\lVert \mathbf{x}_i - \mathbf{x}_j \rVert_2}{\ell}\right)^{\upsilon} k_{\upsilon}\left(\frac{\sqrt{2\upsilon}\lVert \mathbf{x}_i - \mathbf{x}_j \rVert_2}{\ell}\right)$	$\frac{2^{D+\upsilon}\pi^{\frac{D}{2}}\Gamma(\upsilon+\frac{D}{2})\upsilon^{\upsilon}}{\Gamma(\upsilon)\ \ell^{2\upsilon}}\left(\frac{2\upsilon}{\ell^2} + 4\pi^2\lVert\omega\rVert_2^2\right)^{-(\upsilon+\frac{D}{2})}$
Laplacian[a]	$\exp(-\sigma\lVert\mathbf{x}_i - \mathbf{x}_j\rVert_1)$	$\left(\frac{2}{\pi}\right)^{\frac{D}{2}}\prod_{i=1}^{D}\frac{\sigma}{\sigma^2+\omega_i^2}$

[*] $\Gamma(\cdot)$ is the gamma function and $k_\lambda(\cdot)$ is the modified Bessel function of the second kind.

[†] Parameter $\upsilon > 0$. If $\upsilon = 0.5$ the Matén equates to the exponential kernel. As $\upsilon \rightarrow \infty$ the Matén converges to the squared exponential kernel.

[‡] Parameter $\ell > 0$.

[a] Parameter $\sigma > 0$.

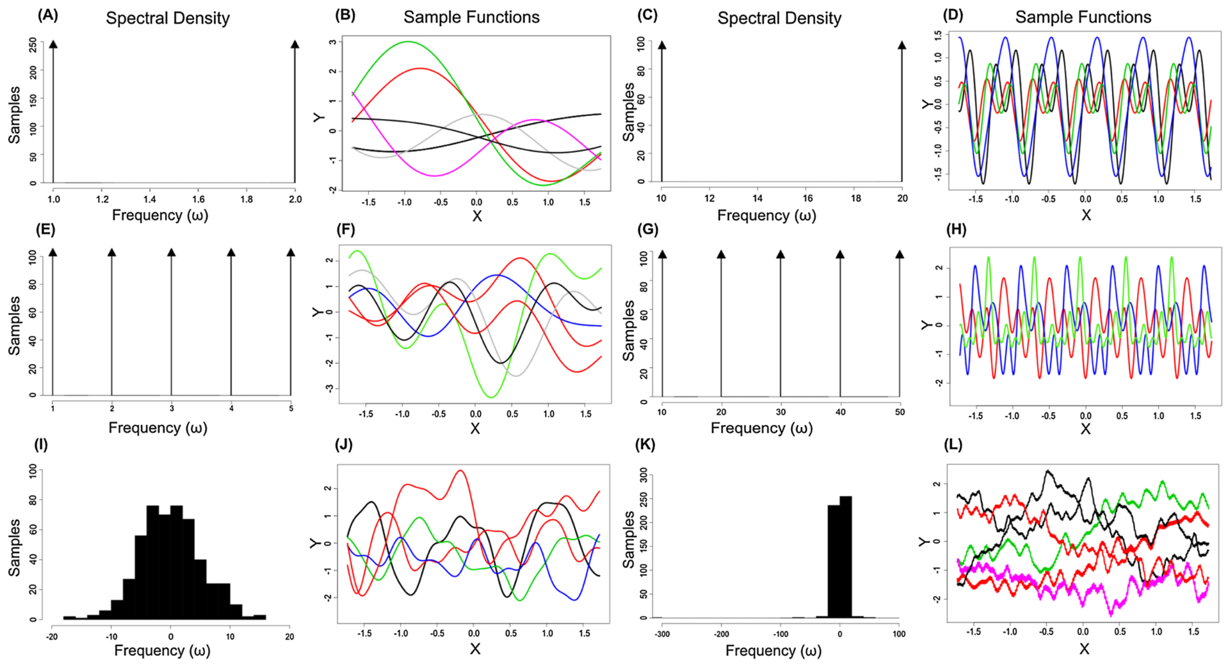

Fig. 2. Power spectral densities (A,C,E,G,I,K) and the functions produced by sampling from the resulting kernel (B,D,F,H,J,L). The spectral densities correspond to sampling from delta functions (arrowheads) such that the sampled frequencies can only take the point values corresponding to each delta. (I) is Gaussian distribution corresponding to a spectral density of the squared exponential kernel. (K) is Cauchy distribution corresponding to a spectral density of the Laplacian kernel.

However, one of the key observations about RFF is that it defines a feature space of its own:

$$\frac{1}{m}\sum_{s=1}^{m}\begin{pmatrix}\cos(\omega_s^T\mathbf{x}_i)\\ \sin(\omega_s^T\mathbf{x}_i)\end{pmatrix}^T\begin{pmatrix}\cos(\omega_s^T\mathbf{x}_j)\\ \sin(\omega_s^T\mathbf{x}_j)\end{pmatrix}=\frac{1}{m}\sum_{s=1}^{m}\Phi_{\text{RFF}}(\mathbf{x}_i)^T\Phi_{\text{RFF}}(\mathbf{x}_j),\qquad \{\omega_s\}_{s=1}^{m}\overset{i.i.d.}{\sim}\mathbb{P}(\omega) \tag{21}$$

where

$$\Phi_{\text{RFF}}(\mathbf{x})=\begin{pmatrix}\cos(\omega_s^T\mathbf{x})\\ \sin(\omega_s^T\mathbf{x})\end{pmatrix},\qquad \{\omega_s\}_{s=1}^{m}\overset{i.i.d.}{\sim}\mathbb{P}(\omega) \tag{22}$$

The data is mapped to a Fourier feature space, Φ_{RFF}, that approximates the previously implicit feature space of the kernel. Technically, Φ_{RFF} is a function space that is dense in a RKHS - the same space of functions from our kernel matrix. Applying this mapping to the entire design matrix is given by $\Phi\text{RFF}(X)=[\cos(X\Omega^T)\sin(X\Omega^T)]\in\mathbb{R}^{N\times 2m}$ where $\Omega\in\mathbb{R}^{m\times d}$ is the frequency matrix with each rows corresponding to a sampled ω $(\omega_1, \ldots, \omega_m)$. Matrix $\Phi\text{RFF}(X)$ is termed the Fourier feature matrix, with each column representing a Fourier feature. See Supplementary Equations 4 for a more comprehensive walk-through of derivation of the Fourier feature matrix from Bochner's theorem.

The Fourier feature matrix contains all the data explicitly mapped into a finite dimensional space that approximates the implicit (and potentially infinite) feature mapping of the kernel function. The explicitly mapped data in the Fourier feature matrix can be used to solve the primal. The primal solution now requires computing and inverting $\Phi_{\text{RFF}}(X)^T\Phi_{\text{RFF}}(X)\in\mathbb{R}^{2m\times 2m}$ with resulting linear model takes the form $\mathbf{y}\sim\Phi_{\text{RFF}}(X)\hat{\mathbf{w}}$. The computational complexity of this primal solution is $O(\text{Nm}^2)$ (where m is the number of $i.i.d.$ samples from $\mathbb{P}(\omega)$), providing a significant computational speed-up when $m < < N$, and is particularly relevant for large datasets. The original paper by Rahimi and Recht (Rahimi and Recht, 2007) showed that every entry in our kernel matrix is approximated to an error of $\pm\,\xi$ with $m=\frac{\log(N)}{\xi^2}$ Fourier features. A more recent result shows that only $\sqrt{N}\log(N)$ features can achieve the same learning bounds as full kernel ridge regression with squared loss (Rudi and Rosasco, 2017). For example, given $N = 100,000$ data points, we would only need $m\approx 3600$ Fourier features to achieve the generalisation error as if we had used all points. However, the full theoretical properties of RFF estimators are still far from fully understood.

9. Variation in the linear model

Throughout this paper, we have focused on squared loss/Gaussian likelihoods, but Fourier features can be used with any loss function. For example, when performing classification with binary data, (i.e. $y\in\{0, 1\}$) the cross-entropy loss (also known as log loss) can be used which is written as:

$$S(\mathbf{w})=-[y\log(\varphi^{-1}(X\mathbf{w}))+(1-y)\log(1-\varphi^{-1}(X\mathbf{w}))] \tag{23}$$

where φ is the Sigmoid or Logit function. The design matrix X can be substituted for the Fourier feature matrix, $\Phi_{\text{RFF}}(X)$ to derive a computationally efficient extension of kernel regression. Other examples include the use of a Poisson likelihood to model count data (Cameron and Trivedi, 2013) or a generalised linear model (GLM) for many other non-Gaussian response variables (Nelder and Wedderburn, 1972). GLMs still use a linear combination of explanatory variables (with a link function controlling how the expected value of the response relates to the linear combination of explanatory variables) and thus can still use feature mapping including RFF to map the model into feature space. The linear model can easily be extended to include uncertainty using Bayesian inference through the inclusion of appropriate likelihood function and priors on the weights (Carlin and Louis, 2008). Many other algorithms use kernels, notably kernel methods, a group of non-parametric, probabilistic machine learning algorithms used for pattern analysis. These include support vector machines for classification or Gaussian processes for regression and can effectively be constructed by applying kernels to Bayesian linear models. These methods work in implicit feature space associated with the kernel and are capable of modelling huge variety of non-linear problems, but also suffer from the "Big N" problem restricting their use to small/medium datasets. Many kernel methods have primal solutions that can be solved by sunstituting in the Fourier feature matrix for a very large computational speed-up.

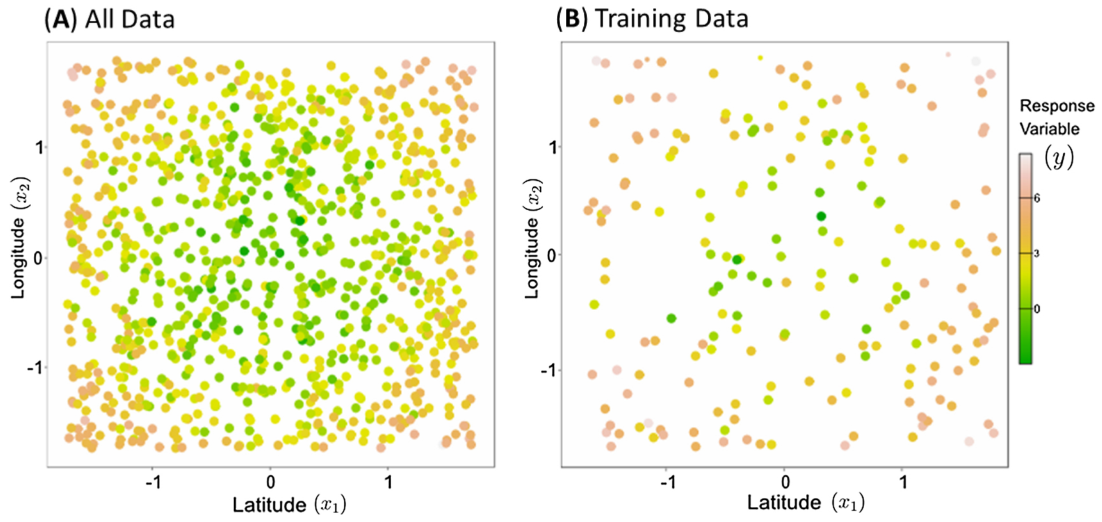

Fig. 3. (A) All 500 points generated from the latent spatial process given by $y_i = x_{i,1}^2 + x_{i,2}^2 + \epsilon_i$ where $\epsilon_i \sim \mathcal{N}(\mu = 0, \sigma^2 = 1)$), and (B) the subset of data points used to train the regression models.

10. Toy example of random Fourier features for spatial analysis

As an example of how to use RFF, we simulate a simple non-linear spatial regression problem. A set of random points in space is generated such that each location has unique coordinates (longitude and latitude). Each location has a response variable generated the function $y = x_1^2 + x_2^2 + \epsilon$ where ϵ is random Gaussian noise ($\epsilon \sim \mathcal{N}(\mu = 0, \sigma^2 = 1)$). Fig. 3A shows 500 random points drawn from the spatial process. The simulated data were used to train three models, a linear regression model, a kernel regression model (no regularisation) and a KRR model. Both the kernel regression and KRR models use 100 Fourier features to approximate a squared exponential kernel. For KRR, k-fold cross-validation was used to find the optimal regularisation parameter (λ). We assume, as is common for nearly all real-world spatial processes, that we only observe a subset of all possible locations. Therefore, the models are trained on only 20% of all the generated points, shown in Fig. 3B. The remaining data not used to training is then used for testing. The code for this example is provided in Supplementary Code 3. The code can easily be changed to any function of the user's choice and the user can specify the number of Fourier features. Note, the provided code generates points at random and therefore a user's results may differ from the exact results shown here.

Each model was trained using the same training data and their predictive performance measured by MSE for both the training and testing data. The training and testing performance of the three models are shown in Table 2. As expected, the non-linear nature of the spatial process results in very poor performance of the linear model with large values of both the training and testing error. The Kernel regression model has excellent training performance, with the infinite feature space of the squared exponential kernel able to capture the non-linear relationship between the spatial coordinates and the response. However, in the absence of regularisation, the kernel regression model greatly overfits the training data resulting in poor testing performance. In comparison, the regularisation applied in kernel ridge regression helps prevent overfitting with the KRR model (with $\lambda = 3.98$) having marginally higher training error than the kernel regression model but less than half the testing error. Therefore, the KRR model trades a small decrease in training performance for a significant increase in generalisability.

The importance regularisation is further illustrated by comparing how the training and testing performance of kernel regression and KRR varies with the number of Fourier features (Fig. 4). Increasing the number of sampled Fourier features increases results in a steady reduction in training error of kernel regression (Fig. 4A, blue line). As the number of features increases the kernel regression model shows increasing testing error (Fig. 4B, blue line) as the model is significantly overfitted to the training data. In comparison, KRR the training error remains constant above 10 features (Fig. 4A, red line) and maintains a stable testing error that is significantly lower than kernel regression even when additional Fourier features are added (Fig. 4B, red line). The KRR prevents overfitting even as more features are added the regularisation by increasing the magnitude of the regularisation parameter, λ (Fig. 4B, inset).

Table 2
Training and testing mean squared error of linear, kernel and Kernel ridge regression models for a basic non-linear spatial problem.

Model	Training error (MSE)	Testing error (MSE)
Linear	2.26	2.73
Kernel Regression[*]	0.64	2.70
Kernel **Ridge** *Regression*[*], [†]	0.88	1.19

[*] Using a squared exponential kernel approximated using the 100 random Fourier features.

[†] Optimal ridge parameter of $\lambda = 3.98$ estimated by k-fold validation.

11. Advanced methods for random Fourier features

11.1. Limitations of RFF

Given the good empirical performance of the RFF method, little has been published on their limitations, including in the context of spatial analysis. Firstly, RFF can be poor at capturing very fine-scale variation as noted in Ton et al. (2018). This is likely due to fine-scale features being captured by the tails of the spectral density that will be infrequently sampled in the Monte Carlo integration. Secondly, from a computational perspective, RFF are very efficient but can still be outpreformed by some state-of-the-art spatial statistics approaches. For example, the sparse matrix approaches based on SPDE as solution GMRF provide impressive savings with complexity $O(m^{1.5})$ (compared to $O(m^3)$ from the RFF primal solution) (Lindgren et al., 2011). Other methods such as the multiresolution Kernel approximation (MRA) provide incredible performance but are only valid for two dimensions (Ding et al., 2017). Thirdly, while the convergence properties of RFF suggest excellent predictive capability (Rudi and Rosasco, 2017), alternative *data-dependent* methods including versions of the Nyström approximation can perform much better in some settings (Yang et al., 2012; Rudi et al., 2015). The following sections will discuss the current methods that address some of these limitations. From here onward, we

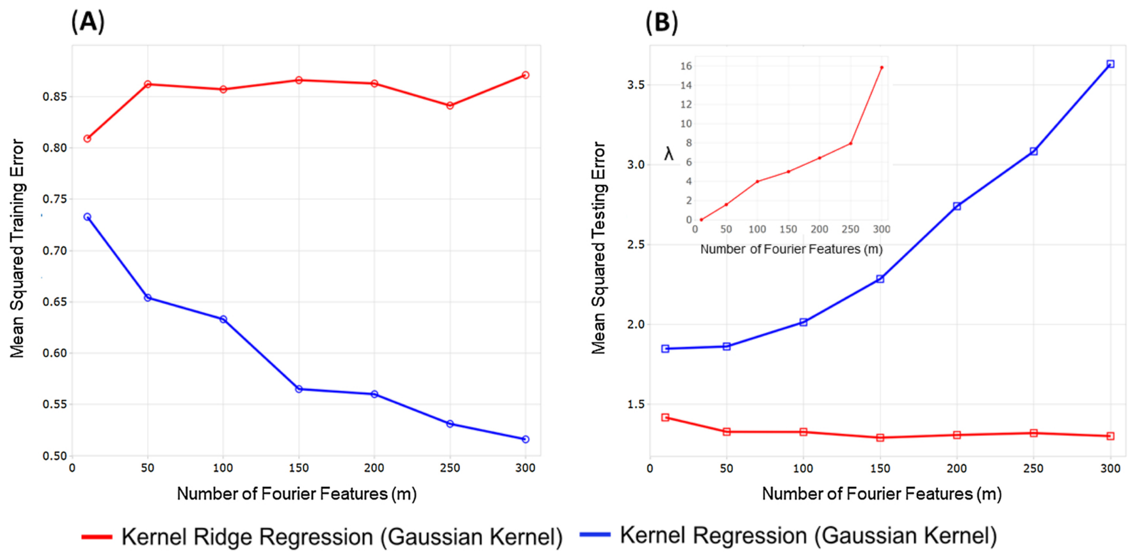

Fig. 4. An example of the bias-variance trade-off for kernel regression (blue) and kernel ridge regression model (red) with a random Fourier feature approximation of a squared exponential kernel. The mean squared error (MSE) is calculated for both the (A) training data and the testing data (B) for an increasing number of sampled Fourier features, with optimal λ (estimated by k-fold cross-validation) for the ridge regression model shown (B *inset*).

call the RFF method described in the previous section as the standard RFF method.

11.2. Quasi-Monte-Carlo Features (QMC RFF)

One of the most significant limitations of standard RFF lies within the Monte Carlo integration. The infinite integral that describes the kernel function is converted to a finite approximation by sampling frequencies from the spectral density (Eq. (20)). The convergence of Monte Carlo integration to the true integral occurs at the rate $O(m^{0.5})$, which means for some problems a large number of features are required to approximate the integral accurately.

A popular alternative is to use Quasi-Monte Carlo (QMC) integration (Avron et al., 2018). In QMC integration, the set of points chosen to approximate the integral is chosen using a deterministic, low-discrepancy sequence (Niederreiter, 1978). In this context, low-discrepancy means the points generated appear random even though they are generated from a deterministic, non-random process. For example, the Halton sequence, that generates points on a uniform hypercube before transforming the points through a quantile function (inverse cumulative distribution function) (Halton, 1964). Low-discrepancy sequences prevent clustering and enforce more uniformity in the sampled frequencies, allowing QMC to converge at close to $O(m^{-1})$ (Asmussen and Glynn, 2007) and can provide substantial improvements in the accuracy of the approximation of the kernel matrix for the same computational complexity. Crucially, QMC is trivial to implement within the RFF framework for some distributions. For example, for the squared exponential kernel, instead of generating features as by taking random samples from a Gaussian, we generate them as:

Code 2 Example of Quasi-Monte Carlo sampling of a Gaussian power spectral density of a squared exponential kernel using a Halton sequence.

11.3. Leverage score sampling

In the standard RFF method, frequencies are sampled with a probability proportional to their spectral density. However, the power spectral density of a kernel does not depend on the data, X (see Table 1). Therefore, the sampling probability of a given frequency is *data-independent*. Data-independent sampling is sub-optimal and can yield very poor approximations (Bach, 2012; Mahoney and Drineas, 2009; Gittens and Mahoney, 2016), and has been identified as one of the reasons RFFs perform poorly in certain situations (Li et al., 2018). An alternative is a *data-dependent* approach, that considers the importance of various features *given* some data. Several data-dependent approaches for RFF have been proposed (Ionescu et al., 2017; Rudi and Rosasco, 2017; Li et al., 2018), but one of the most promising and easiest to implement is sampling from the leverage distribution of the RFF (abbreviated to LRFF) (Li et al., 2018).

Leverage scores are popular across statistics and are a key tool for regression diagnostics and outlier detection (Hoaglin and Welsch, 1978; Velleman and Welsch, 1981). A leverage score measures the *importance* of a given observation on the solution of a regression problem. However, the perspective of leverage scores as a measure of importance can be extended to any matrix. The leverage scores of a matrix A is given by the diagonal matrix $T = A(AA^T)^{-1}A^T$ with leverage score for ith row of matrix A is equal to the ith diagonal element in matrix T, denoted as $\tau_{i,i}$ and calculated by:

$$\tau_{i,i} = \mathbf{a}_i^T(AA^T)^{-1}\mathbf{a}_i = [A(AA^T)^{-1}A^T]_{ii} \tag{24}$$

$\tau_{i,i}$ can also be seen as a measure of the importance of the row $\mathbf{a}_i$.

Most leverage score sampling methods apply ridge regularisation to the leverage scores equation, controlled by the regularisation parameter λ_{LRFF} given by:

$$\tau_{i,i}(\lambda) = \mathbf{a}_i^T(AA^T + \lambda_{\text{LRFF}}I)^{-1}\mathbf{a}_i \tag{25}$$

```
#X = data matrix of dimensions (N x d), m = number of sampled frequencies
library(randtoolbox) # Package required for generating Halton sequence
Omega = matrix(qnorm((halton(m,ncol(X)))),m) # quasi-MC of a Gaussian
Proj = X %*% t(Omega) # projection
Phi = cbind(cos(Proj), sin(Proj)) / sqrt(m) # Fourier features
K = Phi %*% t(Phi) # kernel matrix
```

The resulting scores are termed ridge leverage scores (El Alaoui and Mahoney, 2014). The regularisation serves a nearly identical purpose as when applied in the context of linear regression; ensuring the matrix inversion is always possible with scores that less sensitive to perturbations in the underlying matrix. The addition of the ridge regularisation stabilises leverage scores computation and permits fast leverage score sampling methods that approximate leverage scores using subsets of the full data (Musco and Musco, 2017; Rudi et al., 2018; Drineas et al., 2012; Cohen et al., 2017).

Leverage scores improve upon the standard RFF method by sampling ω's with probability proportional to importance rather than their spectral density. The important features are extracted by sub-sampling columns of a matrix $\Phi_{\text{RFF}}(\mathbf{X})$ with probability proportional to their leverage scores. The resulting sub-sample should contain all the important features such that an equally accurate approximation to the full matrix can be achieved using a smaller number of features. A key distinction to note is that the formula for the leverage score in Eq. (25), and much of the literature, seeks the leverage scores of the rows ($\tau_{i,i}$ give the importance of the ith row of A). For RFF, we want the leverage scores of the columns as they correspond to the Fourier features, and thus the transpositions in Eq. (25) must be swapped. The ridge leverage score of the Fourier features matrix is given by:

$$\begin{aligned} T(\lambda) &= \text{diag}(\Phi(X)(\Phi(X)^T\Phi(X) + \lambda_{\text{LRFF}}I)^{-1}\Phi(X)^T) \\ \tau_{i,i}(\lambda) &= \Phi(\mathbf{x}_i)(\Phi(X)^T\Phi(X) + \lambda_{\text{LRFF}}I)^{-1}\Phi(\mathbf{x}_i)^T \end{aligned} \quad (26)$$

Note, in Eq. (26) RFF subscript is suppressed for legibility ($\Phi(X) = \Phi_{\text{RFF}}(X)$) and $\Phi(\mathbf{x}_i) = \Phi_{\text{RFF}}(\mathbf{x}_i)$. The computation and inversion of the matrix $\Phi_{\text{RFF}}(\mathbf{X})$ increases the computational burden of this approach, but only has to be calculated once for a given regularisation parameter. With suitable scaling, we can now sample Fourier features with a probability proportional to the leverage distribution, allowing us to sample features proportional to their importance. The code is given by:

Code 3 LRFF example.

```
#X = data matrix of dimensions (N x d)
# Generate the m frequency (Omega) from the PSD
Proj = X %*% t(Omega)    # Projection
Phi = cbind(cos(Proj), sin(Proj)) / sqrt(m) # Fourier features
M = t(Phi) %*% Phi  # Primal matrix
T_lrff <- M %*% solve(M + n*diag(x=lambda,m))
pi_s <- diag(T_lrff) # Diagonal elements
l <- sum(diag(T_lrff))  # Sum of diagonal elements of T
is_wgt <- sqrt(pi_s/l)  # Leverage scores
```

11.4. Orthogonal random features

One of the benefits of RFF is that they allow us to define kernels in higher dimensions. For example, one can use a kernel in 4-dimensions to represent Cartesian spatial coordinates x, y, z and time, t, the foundation for any spatiotemporal modelling. However, increasing dimensionality comes at a cost of increased variance of the RFF estimate and significantly more features to achieve an accurate approximation (Felix et al., 2016). One proposed approach to solve this issue with high dimensional kernels is to draw each new feature dimension orthogonally.

In the standard RFF method, the sampled frequencies can be concatenated into a frequency matrix, $\Omega \in \mathbb{R}^{m\times d}$. If we consider a squared exponential kernel, Ω is actually just a random Gaussian matrix, as sampled frequencies are simply standard normal distribution random variables scaled by the kernel parameter, σ. Therefore, the matrix $\Omega = \frac{1}{\sigma}G$, where G is a random Gaussian matrix of dimension $\mathbb{R}^{m\times d}$. In orthogonal random features (ORF), the aim is to impose orthogonality on Ω, such that it contains significantly less redundancy than a random Gaussian matrix, capable of faster convergence to the full kernel matrix with lower variance estimates. The G in this section refers to the random Gaussian matrix and should not be confused with the gram matrix, but the G notation is used to be consistent with the original paper (Felix et al., 2016).

The simplest method to impose orthogonality would be to replace G with a random orthogonal matrix, O. However, simply replacing G with O means that the RFF will no longer be an unbiased estimator (Rahimi and Recht, 2007). For example, for the squared exponential/Gaussian kernel, the row norms of the G matrix will follow a Chi distribution. In comparison, the orthogonal matrix, O, will have rows with unit norm by definition. Thus, replacing G with O does not produce an equivalent unbiased estimator. To return to equivalency, the orthogonal matrix, O, must be scaled by the diagonal matrix, S, with diagonal entries that are random variables drawn from a Chi distribution with D degrees of freedom. This ensures that the row norms of G and SO will be identically distributed and can be used to construct a matrix of orthogonal random frequencies given by $\Omega_{\text{ORF}} = \frac{1}{\sigma}\text{SO}$. The elements of S are only Chi distributed random variables when G is a random Gaussian matrix. For other kernels, the diagonal elements of matrix S are computed as the L2 norm for the corresponding row in G (the definition of a Chi distributed random variable is identical to taking the L2 norm of a set of standard normal distributed variables). Therefore, the ith diagonal element of S is calculated by:

$$s_{i,i} = \|G_i\|_2 = \sqrt{\sum_{j=1}^{D} G_{i,j}^2} \quad (27)$$

Thus, generating orthogonal random features for a given kernel requires three steps. First, derive the orthogonal matrix O by performing QR decomposition on the feature matrix G (where O corresponding to the Q matrix of QR decomposition). See Gentle (2012) for an excellent summary of QR decomposition. Second, compute the entries of the diagonal matrix, S, by talking the L2 norms of corresponding rows of G. Finally, compute the orthogonal feature matrix as $\Omega_{\text{ORF}} = \frac{1}{\sigma}\text{SO}$. This orthogonal frequency matrix replaces the random frequency matrix to generate the orthogonal random feature matrix $\Phi_{\text{ORF}}(X)$, computed as $\Phi_{\text{ORF}}(X) = [\cos(X\Omega_{\text{ORF}}^T)\ \sin(X\Omega_{\text{ORF}}^T)] \in \mathbb{R}^{N\times 2m}$. The matrix $\Phi_{\text{ORF}}(X)$ will contain orthogonal Fourier features with significantly less redundant features than the standard RFF method. The R code for ORF is as follows:

Code 4 ORF example.

```r
# Generate the m x d orthogonal frequency matrix
omege <- c()
G <- matrix(rnorm(m*d),nrow=m,ncol=d) # Random Gaussian Matrix
O <- qr.Q(qr(G), complete = T) # Q matrix from QR Decompostion
S <- diag(sqrt(rowSums(G**2))) # Diagonal Matrix
omega <- S%*%G
```

Felix et al. (2016) extended the ORF method further to a method known as *structured* ORF (SORF). The SORF method avoids the computationally expensive steps of deriving the orthogonal matrix ($O(N^3)$ time) and computing random basis matrix ($O(N^2)$ time) by replacing the random orthogonal matrix, O, by a class of specially structured matrices (consisting of products of binary diagonal matrices and Walsh-Hadamard matrices) that have orthogonality with near-Gaussian entries and can use highly efficient algorithms (such as the fast Walsh-Hadamard transform) (Felix et al., 2016). The SORF method maintains a lower approximation error than the standard RFF method and is significantly more computationally efficient than ORF, with computing $\Phi_{\text{SORF}}(X)$ taking only $O(\text{N}\log(N))$ time. However, technically the resulting approximation of the kernel is no longer unbiased.

11.5. Non-stationary and arbitrary kernel functions

One of the most significant limitations to the standard RFF method is the restriction to shift-invariant kernels, where $k(\mathbf{x}_i, \mathbf{x}_j) = k(\mathbf{x}_i - \mathbf{x}_j)$. This restriction means that the kernel value is only dependent on the lag or distance between the points rather than the actual locations. This property imposes stationarity on the spatiotemporal process. While this assumption is not unreasonable, and non-stationarity is often unidentifiable, in some cases the relaxation of stationarity can significantly improve model performance (Paciorek and Schervish, 2006).

To extend the RFF method to non-stationary kernels requires a more general representation of Bochner's theorem capable of capturing the spectral characteristics of both stationary and non-stationary kernels. This extension (Yaglom, 1987) states than any kernel (stationary or non-stationary) can be expressed as its Fourier transform in the form of:

$$k(\mathbf{x}_i, \mathbf{x}_j) = \int_{R^d \times R^d} e^{i(\omega_1^T \mathbf{x}_i - \omega_2^T \mathbf{x}_j)} \mathbb{P}(\omega_1)\mathbb{P}(\omega_2)\ d\omega_1 d\omega_2 \tag{28}$$

This equation is nearly identical to the original derivation of Bochner's theorem given in Eq. (18), but now we have two spectral densities on $\mathbb{R}^D$ to integrate over. It is easy if that if the two spectral densities are the same, the function returns to the definition for a stationary kernel. Applying the same treatment and Monte Carlo integration can be performed to give the feature space of the non-stationary kernel (Ton et al., 2018), now given by:

$$\Phi_{\text{RFF}}(\mathbf{x}) = \begin{pmatrix} \cos(\omega_1^T \mathbf{x}) + \cos(\omega_2^T \mathbf{x}) \\ \sin(\omega_1^T \mathbf{x}) + \sin(\omega_2^T \mathbf{x}) \end{pmatrix}, \quad \omega_{\{1,2\}} = \{\omega_{\{1,2\},1}, \dots, \omega_{\{1,2\},m}\} \overset{i.i.d.}{\sim} \mathbb{P}_{\{1,2\}}(\omega) \tag{29}$$

Note, that this derivation requires drawing independent samples for both of the spectral densities, $\mathbb{P}_l(\omega)$, such that we generate two frequency matrices, $\Omega_l \in \mathbb{R}^{m \times d}$.

In both the stationary and non-stationary case the choice of the kernel is often arbitrary or made with knowledge of the process being modelled. For example, if the spatial data is expected to be very smooth, then a squared exponential kernel can be used. It is, however, possible to treat ω as unknown kernel parameters variables and infer their values (Ton et al., 2018). This is equivalent to deriving an empirical spectral distribution. This strategy is data dependent and can achieve impressive results; however, great care must be taken to avoid overfitting (Ton et al., 2018).

12. Conclusion

Regression is a key technique for nearly all scientific disciplines and can be extended from its simplest forms to highly complex and flexible models capable of describing nearly any type of data using feature mapping. Kernels permit working with infinite dimensional feature spaces but are not computationally feasible with large datasets. The RFF method is capable of approximating the full kernel matrix and valid for kernels in high dimensions, but its key advantage over other methods is that using RFF to solve the primal rather than the dual solution provides significant computational benefit. All these advantages are achieved using only a probability distributions and trigonometric functions and can be encapsulated in 4 lines of code. While some advanced derivatives of existing sparse or low-rank methods may outperform the standard RFF, several advanced RFF methods are emerging that continue to improve on the standard RFF method. To that end, random Fourier features and their extensions represent an exciting new tool for multi-dimensional spatial analysis on large datasets.

Author contributions

PM drafted the manuscript which was read, revised for critical intellectual content, modified and then approved by all authors. HC provided significant support on the mathematical aspects, and EG provided significant support on the spatial statistics. SB conceived and the paper and provided significant support on the structuring, drafting and editing of the paper itself.

Funding

PM would like to acknowledge the Medical Research Council-Doctoral Training Programme at Imperial College London. HC's research is funded by a studentship from the Wellcome Trust. SB receives financial support from the Bill and Melinda Gates Foundation (1606H5002/JH6). EG is supported by an MRC Strategic Skills Fellowship in Biostatistics (MR/M015297/1). The funders had no role preparation or publication of the manuscript. The views, opinions, assumptions or any other information set out in this article are solely those of the authors. The authors thank the UK National Institute for Health Research Health Protection Research Unit (NIHR HPRU) in Modelling Methodology at Imperial College London in partnership with Public Health England (PHE) for funding (grant HPRU-2012-10080).

Conflicts of interest

None declared.

Acknowledgements

We gratefully acknowledge Reviewer 1 and 2 for their insightful comments in the manuscript review process.

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.epidem.2019.100362.

References

Andres, L.A., et al., 2018. Geo-Spatial Modeling of Access to Water and Sanitation in Nigeria. The World Bank.

Asmussen, S., Glynn, P.W., 2007. Stochastic Simulation: Algorithms and Analysis. Springer Science & Business Media.

Avron, H., et al., 2018. Random Fourier Features for Kernel Ridge Regression: Approximation Bounds and Statistical Guarantees. CoRR abs/1804.09893. arXiv: 1804.09893.

Bach, F.R., Jordan, M.I., 2005. Predictive low-rank decomposition for kernel methods. In: Proceedings of the 22nd international conference on Machine learning – ICML'05. ISBN: 1595931805.

Bach, F.R., 2012. Sharp Analysis of Low-Rank Kernel Matrix Approximations. CoRR abs/1208.2015. arXiv: 1208.2015.

Bell, J.B., Tikhonov, A.N., Arsenin, V.Y., 1978. Solutions of ill-posed problems. Math. Comput ISSN: 00255718.

Bochner, S., Chandrasekharan, K., 1949. Fourier Transforms. Princeton University Press.

Boyd, S., Vandenberghe, L., 2004. Convex Optimization. Cambridge University Press.

Bracewell, R.N., Bracewell, R.N., 1986. The Fourier Transform and Its Applications. McGraw-Hill, New York.

Cameron, A.C., Trivedi, P.K., 2013. Regression Analysis of Count Data. Cambridge University Press.

Carlin, B.P., Louis, T.A., 2008. Bayesian Methods for Data Analysis. CRC Press.

Cohen, M.B., Musco, C., Musco, C., 2017. Input sparsity time low-rank approximation via ridge leverage score sampling. Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms 1758–1777.

Cuadros, D.F., et al., 2017. Mapping the spatial variability of HIV infection in Sub-Saharan Africa: effective information for localized HIV prevention and control. Sci. Rep. 7, 9093.

Davidson, R., MacKinnon, J.G., et al., 2004. Econometric Theory and Methods. Oxford University Press, New York.

Diggle, P., Ribeiro, P., 2007. Model-Based Geostatistics. English. Springer ISBN: 0387329072 978-0387329079.

Diggle, P.J., Tawn, J., Moyeed, R., 1998. Model-based geostatistics. J. R. Stat. Soc. Ser. C (Appl. Stat.) 47, 299–350.

Ding, Y., Kondor, R., Eskreis-Winkler, J., 2017. Multiresolution Kernel Approximation for Gaussian Process Regression in Advances in Neural Information Processing Systems. pp. 3740–3748.

Domingos, P., 2000. A Unified Bias-Variance Decomposition and its Applications. Science.

Drineas, P., Magdon-Ismail, M., Mahoney, M.W., Woodruff, D.P., 2012. Fast approximation of matrix coherence and statistical leverage. J. Mach. Learn. Res. 13, 3475–3506.

El Alaoui, A., Mahoney, M., 2014. Fast Randomized Kernel Methods With Statistical Guarantees, vol. 1411 arXiv preprint. arXiv.

Farrar, D.E., Glauber, R.R., 1967. Multicollinearity in regression analysis: the problem revisited. Rev. Econ. Stat. 92–107.

Felix, X.Y., Suresh, A.T., Choromanski, K.M., Holtmann-Rice, D.N., Kumar, S., 2016. Orthogonal Random Features in Advances in Neural Information Processing Systems. pp. 1975–1983.

Geman, S., Bienenstock, E., Doursat, R., 1992. Neural networks and the bias/variance dilemma. Neural Comput. ISSN: 0899-7667. arXiv: arXiv:1011.1669v3.

Gentle, J.E., 2012. Numerical Linear Algebra for Applications in Statistics. Springer Science & Business Media.

Gething, P.W., et al., 2016. Mapping Plasmodium falciparum mortality in Africa between 1990 and 2015. New Engl. J. Med. 375, 2435–2445.

Gittens, A., Mahoney, M.W., 2016. Revisiting the Nyström method for improved large-scale machine learning. J. Mach. Learn. Res. 17, 3977–4041.

Gleason, B.L., et al., 2017. Geospatial analysis of household spread of Ebola virus in a quarantined village-Sierra Leone, 2014. Epidemiol. Infect. 145, 2921–2929.

Graetz, N., et al., 2018. Mapping local variation in educational attainment across Africa. Nature 555, 48.

Halton, J.H., 1964. Algorithm 247: radical-inverse quasi-random point sequence. Commun. ACM 7, 701–702. https://doi.org/10.1145/355588.365104. ISSN: 0001-0782.

Hay, S.I., et al., 2013. Global mapping of infectious disease. Philos. Trans. R. Soc. B: Biol. Sci. 368, 20120250.

Hoaglin, D.C., Welsch, R.E., 1978. The hat matrix in regression and ANOVA. Am. Stat. 32, 17–22.

Hoerl, A.E., Kennard, R.W., 1970. Ridge regression: biased estimation for nonorthogonal problems. Technometrics. ISSN: 15372723. arXiv: 9809069v1 [arXiv:gr-qc].

Ionescu, C., Popa, A., Sminchisescu, C., 2017. Large-scale data-dependent kernel approximation. In: In: Singh, A., Zhu, J. (Eds.), Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, vol. 54. PMLR, Fort Lauderdale, FL, USA, pp. 19–27 April.

Josepha, G., Gething, P.W., Bhatt, S., Ayling, S.C., 2019. Understanding the Geographical Distribution of Stunting in Tanzania: A Geospatial Analysis of the 2015–16. Demographic and Health Survey.

Kanagawa, M., Hennig, P., Sejdinovic, D., Sriperumbudur, B.K., 2018. Gaussian Processes and Kernel Methods: A Review on Connections and Equivalences. arXiv preprint arXiv:1807.02582.

Li, Z., Ton, J.-F., Oglic, D., Sejdinovic, D., 2018. A Unified Analysis of Random Fourier Features. arXiv preprint arXiv:1806.09178.

Lindgren, F., Rue, H., Lindström, J., 2011. An explicit link between Gaussian fields and Gaussian Markov random fields: the stochastic partial differential equation approach. J. R. Stat. Soc.: Ser. B (Stat. Methodol.) 73, 423–498. https://doi.org/10.1111/j.1467-9868.2011.00777.x.

Mahoney, M.W., Drineas, P., 2009. CUR matrix decompositions for improved data analysis. Proc. Natl. Acad. Sci pnas-0803205106.

McCullagh, P., Nelder, J.A., 1989. Generalized Linear Models. Chapman and Hall, London, UK.

Mena, I., et al., 2016. Origins of the 2009 H1N1 influenza pandemic in swine in Mexico. Elife 5, e16777.

Musco, C., Musco, C., 2017. Recursive Sampling for the Nystrom Method in Advances in Neural Information Processing Systems. pp. 3833–3845.

Nelder, J.A., Wedderburn, R.W., 1972. Generalized linear models. J. R. Stat. Soc.: Ser. A (Gen.) 135, 370–384.

Niederreiter, H., 1978. Quasi-Monte Carlo methods and pseudo-random numbers. Bull. Am. Math. Soc. 84, 957–1041.

Noma, M., et al., 2002. Rapid epidemiological mapping of onchocerciasis (REMO): its application by the African Programme for Onchocerciasis Control (APOC). Ann. Trop. Med. Parasitol. 96, S29–S39. https://doi.org/10.1179/000349802125000637. ISSN: 0003-4983.

Osgood-Zimmerman, A., et al., 2018. Mapping child growth failure in Africa between 2000 and 2015. Nature 555, 41.

Paciorek, C.J., Schervish, M.J., 2006. Spatial modelling using a new class of nonstationary covariance functions. Environmetrics 17, 483–506.

Rahimi, A., Recht, B., 2007. Random features for large scale kernel machines. Adv. Neural Inf. Process. Syst. ISSN: 0033-6599, arXiv: arXiv:1409.1151v1.

Rasmussen, C.E., Williams, C.K.I., 2005. Gaussian Processes for Machine Learning (Adaptive Computation and Machine Learning). The MIT Press ISBN: 026218253X.

Rudi, A., Rosasco, L., 2017. Generalization Properties of Learning With Random Features in Advances in Neural Information Processing Systems. pp. 3215–3225.

Rudi, A., Camoriano, R., Rosasco, L., 2015. Less is More: Nyström Computational Regularization in Advances in Neural Information Processing Systems. pp. 1657–1665.

Rudi, A., Calandriello, D., Carratino, L., Rosasco, L., 2018. On Fast Leverage Score Sampling and Optimal Learning in Advances in Neural Information Processing Systems. pp. 5673–5683.

Rudin, W., 1990. Fourier Analysis On Groups. ISBN: 047152364X.

Rue, H., Held, L., 2005. Gaussian Markov Random Fields: Theory and Applications. CRC Press.

Saad, Y., Schultz, M.H., 1986. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. SIAM J. Sci. Stat. Comput. 7, 856–869.

Shawe-Taylor, J., Cristianini, N., et al., 2004. Kernel Methods for Pattern Analysis. Cambridge University Press.

Stein, M.L., 2012. Interpolation of Spatial Data: Some Theory for Kriging. Springer Science & Business Media.

Straeter, T.A., 1971. On the Extension of the Davidon-Broyden Class of Rank One, Quasi-Newton Minimization Methods to an Infinite Dimensional Hilbert Space With Applications to Optimal Control Problems.

Tatem, A.J., et al., 2010. Ranking of elimination feasibility between malaria-endemic countries. Lancet 376, 1579–1591.

Tikhonov, A.N., 1963. Solution of incorrectly formulated problems and the regularization method. Soviet Math ISSN: 10634584.

Ton, J.-F., Flaxman, S., Sejdinovic, D., Bhatt, S., 2018. Spatial mapping with Gaussian processes and nonstationary Fourier features. Spat. Stat. 28, 59–78. One world, one health, ISSN: 2211-6753. http://www.sciencedirect.com/science/article/pii/S2211675317302890.

Vatcheva, K.P., Lee, M., McCormick, J.B., Rahbar, M.H., 2016. Multicollinearity in regression analyses conducted in epidemiologic studies. Epidemiology (Sunnyvale, Calif.) 6.

Velleman, P.F., Welsch, R.E., 1981. Efficient computing of regression diagnostics. Am. Stat. 35, 234–242.

Whittle, P., 1954. On stationary processes in the plane. Biometrika 434–449.

Whittle, P., 1963. Stochastic-processes in several dimensions. Bull. Int. Stat. Inst. 40, 974–994.

Williams, C.K.I., Seeger, M., 2001. In: In: Leen, T.K., Dietterich, T.G., Tresp, V. (Eds.), Advances in Neural Information Processing Systems, vol. 13. MIT Press, pp. 682–688. http://papers.nips.cc/paper/1866-using-the-nystrommethod-to-speed-up-kernel-machines.pdf.

Yaglom, A.M., 1987. Correlation Theory of Stationary and Related Random Functions. Springer New York, New York, NY.

Yang, T., Li, Y.-f., Mahdavi, M., Jin, R., Zhou, Z.-H., 2012. In: In: Pereira, F., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (Eds.), Advances in Neural Information Processing Systems, vol. 25. Curran Associates, Inc., pp. 476–484. http://papers.nips.cc/paper/4588-nystrom-method-vs-random-fourier-features-a-theoretical-andempirical-comparison.pdf.