# Hilbert maps: scalable continuous occupancy mapping with stochastic gradient descent

Fabio Ramos and Lionel Ott
School of Information Technologies, The University of Sydney, Australia.

***Abstract*—The vast amount of data robots can capture today motivates the development of fast and scalable statistical tools to model the environment the robot operates in. We devise a new technique for environment representation through continuous occupancy mapping that improves on the popular occupancy grip maps in two fundamental aspects: 1) it does not assume an *a priori* discretisation of the world into grid cells and therefore can provide maps at an arbitrary resolution; 2) it captures statistical relationships between measurements naturally, thus being more robust to outliers and possessing better generalisation performance. The technique, named Hilbert maps, is based on the computation of fast kernel approximations that project the data in a Hilbert space where a logistic regression classifier is learnt. We show that this approach allows for efficient stochastic gradient optimisation where each measurement is only processed once during learning in an online manner. We present results with three types of approximations, Random Fourier, Nyström and a novel sparse projection. We also show how to extend the approach to accept probability distributions as inputs, i.e. when there is uncertainty over the position of laser scans due to sensor or localisation errors. Experiments demonstrate the benefits of the approach in popular benchmark datasets with several thousand laser scans.**

## I. Introduction

Representing the physical properties of 3D space is central to robotics, from manipulation and grasping to autonomous navigation. Amongst the many physical properties characterising the environment the likelihood that a particular point is occupied by a solid object which the robot needs to interact with is certainly one of the most important. Traditional techniques to create a map of occupancy rely on the discretisation of an area into regular sized cells to form a fixed grid on which a sensor model or likelihood function is applied to estimate the posterior of occupancy given some sensory data, for example, laser scans or sonars [4, 5]. One of the main limitations of such techniques is the assumption that each cell in the grid is independent of each other and the posterior computation for the entire map is performed separately for each cell. This assumption disregards important spatial relationships between cells and leads to maps with a series of "gaps" between cells. For example, cells with no observations have a 0.5 likelihood of being occupied even though they are next to cells with high likelihood of being occupied. The problem becomes more severe in 3D maps where the number of cells necessary to represent the environment with the same resolution grows exponentially as does the number of required observations. In indoor environments when the area to be mapped is relatively small and the density of observations is large, occupancy grids are generally sufficient in providing a representation that is both fast and compact. However, representing large 3D outdoor regions with sparse observations still remains a challenge.

In an attempt to resolve some of these issues, the Gaussian processes occupancy map (GPOM) [11, 10] was proposed. The idea is to place a Gaussian prior over the space of functions mapping locations to the occupancy class. The method is continuous, i.e., it does not require a prior discretisation of the space and nonparametric; the complexity of the representation grows with the number of data points. The final Gaussian process classifier model possesses many of the advantages we would like to have in a spatial representation as it directly captures spatial relationships through a parametrised covariance function and produces principled probabilistic posteriors naturally encoding the uncertainty of the process. The main drawback is the computational complexity that without approximations or division of the data into smaller sets scales cubically with the size of the data.

We propose a simpler and faster approach to continuous occupancy mapping in this paper. By utilising recent advancements in optimisation [25] and efficient kernel approximations, we represent the occupancy property of the world with a linear discriminative model operating on a high-dimension feature vector that projects observations into a reproducing kernel Hilbert space (RKHS) [15]. The objective function for training the model is convex in the parameters and therefore the global optimum can be found. Furthermore, the model can be trained and updated using *stochastic gradient descent* making the computation theoretically independent of the number of observations. The key to our approach is to quickly generate a large number of features whose dot product approximate the well-known *radial basis function kernel* (RBF) [15]. The RBF kernel can be seen as a feature mapping into a infinite dimensional space that can asymptotically represent the complexity of the physical world. We present three solutions to approximate the kernel: 1) The first is based on the recently proposed *Random Kitchen Sinks* by [13, 14]; The second is based on the Nyström approximation which is very popular in kernel machines [22]; 3) Finally we introduce a novel feature mapping that generates sparse features better capturing local information. We also show how to generalise these features to accept probability distributions as inputs which are more robust to handle localisation errors or sensor noise. As opposed to GPOM, our method can be updated in linear time and scales well with large amounts of data.

The technical contributions of the paper are:

1) Hilbert maps; a novel continuous occupancy map technique scalable to large datasets and updated in linear

time;

2) A novel sparse Hilbert space feature that better preserves local information and leads to faster stochastic gradient descent iterations;
3) A generalisation of the method to receive probability distributions as inputs to accommodate, in a principled manner, the uncertainty in the position of the measurements.

The paper is organised as follows. We first introduce the method, the features and the extensions to probabilistic inputs in Section II. The objective function and optimisation for online learning through stochastic gradient descent is introduced in Section III. We discuss relationships between Hilbert maps, GPOMs and recent results in machine learning related to important aspects of this technique in Section IV. Experiments on benchmark datasets and comparisons are presented in Section V, and conclusions with ideas for future work are in Section VI.

## II. Hilbert maps

We begin the presentation of the method by first introducing notation. We assume a robot captures a dataset $\mathcal{D} = \{\mathbf{x}_i, y_i\}_{i=1}^N$, where $\mathbf{x_i} \in \mathbb{R}^D$ is a point in $2D$ or $3D$ space and $y_i \in \{-1, +1\}$ is a categorical variable corresponding to the occupancy property of $\mathbf{x}_i$. The dataset is obtained while the robot moves in the environment with a range sensor such as a laser scanner. Randomly selected points in the line segment of a laser beam between the sensor and an object are labelled as unoccupied. The final point in the beam generating a return is labelled as occupied. The length of the beam determines the number of unoccupied points. Due to the amount of beams in laser scans, one point for every one to two metres of beam length is typically sufficient. Selecting the position along the beam at random creates a more uniformly distributed dataset over the free space compared to fixed distance interval sampling. We also assume that the dataset is incrementally built as the robot collects more data as it moves in the environment.

Given the dataset, our objective is to incrementally learn a discriminative model $p(y|\mathbf{x}, \mathbf{w})$ parametrised by a vector $\mathbf{w}$ to predict the occupancy property for new query points $\mathbf{x}_*$. In this work we adopt a very simple *logistic regression classifier* (LR) that is simple and fast to learn while being directly amenable to online learning through stochastic gradient descent (SGD). The probability of nonoccupancy for a point $\mathbf{x}_*$ can be easily computed as:

$$p(y_* = -1|\mathbf{x}_*, \mathbf{w}) = \frac{1}{1+\exp(\mathbf{w}^T\mathbf{x}_*)}, \tag{1}$$

while $p(y_* = +1|\mathbf{x}_*, \mathbf{w}) = 1 - p(y_* = -1|\mathbf{x}_*, \mathbf{w})$ is the probability of occupancy.

The model can be seen as the sigmoid *logit* function applied to a linear projection of the input $\mathbf{x}$. However, how can this simple linear model be able to represent the complexity of the physical world? The key to this problem is to apply the discriminative model not directly to the inputs $\mathbf{x}$ but to a large number of features computed from $\mathbf{x}$, denoted as $\hat{\Phi}(\mathbf{x})^1$. As we shall see next, the dot product of these features can approximate popular kernels commonly used in kernel machines for nonlinear classification. These kernels define a Hilbert space and can represent a nonlinear mapping of the inputs to a space of potentially infinite dimension (for example in the case of the RBF kernel [8]) with sufficient complexity to represent the environment. Note, however, that the advantage of using the feature approximation to the kernel rather than the kernel itself is that we can learn the model using fast primal procedures rather than expensive quadratic programming approaches as demonstrated in [16] for the case of support vector machines.

In the next three sections, we show approaches to generate features $\hat{\Phi}(\cdot)$ that efficiently approximate particular kernels; $k(\mathbf{x}, \mathbf{x}') \approx \hat{\Phi}(\mathbf{x})^T\hat{\Phi}(\mathbf{x}')$.

### A. Random Fourier features

This kernel approximation method is based on the work of Rahimi and Recht [13] with approximation bounds presented in [14] for general learning problems. Formally, a kernel $k(\mathbf{x}, \mathbf{x}')$ defines a Hilbert space with inner product $\langle\cdot,\cdot\rangle$ from a feature vector $\Phi(\mathbf{x})$ such that

$$\forall \mathbf{x}, \mathbf{x}' \in \mathbb{R}^D : k(\mathbf{x}, \mathbf{x}') = \langle\Phi(\mathbf{x}), \Phi(\mathbf{x}')\rangle. \tag{2}$$

If a kernel is shift invariant (also called stationary) it can be written as $k(\tau)$ where $\tau = \mathbf{x} - \mathbf{x}'$ and Bochner's Theorem [6] can be applied to create a representation in terms of its Fourier transform.

**Theorem 1** *(Bochner's Theorem) Any shift invariant kernel $k(\tau)$, $\tau \in \mathbb{R}^D$, with a positive finite measure $d\mu(\mathbf{s})$ can be represented in terms of its Fourier transform as,*

$$k(\tau) = \int_{\mathbb{R}^D} e^{-i\mathbf{s}\cdot\tau} d\mu(\mathbf{s}). \tag{3}$$

The proof can be found in [6]. If $\mu$ has a density $S(\mathbf{s})$, the measure $d\mu(\mathbf{s})$ can be represented as $S(\mathbf{s})d\mathbf{s} = d\mu(\mathbf{s})$ and $S(\mathbf{s})$ is called the *spectral density* of $k$. We can then write

$$k(\tau) = \int_{\mathbb{R}^D} e^{-i\mathbf{s}\cdot\tau} S(\mathbf{s}) d\mathbf{s} = E_{S(\mathbf{s})}\left[e^{-i\mathbf{s}\cdot\tau}\right],$$

where $E_{S(\mathbf{s})}[\cdot]$ denotes the expectation w.r.t. the density $S(\mathbf{s})$. The expected value can thus be approximated as

$$k(\tau) \approx \frac{1}{n}\sum_{k=1}^{n} e^{-i\mathbf{s_k}\cdot\tau} = \langle\hat{\Phi}(\mathbf{x}), \hat{\Phi}(\mathbf{x}')\rangle, \tag{4}$$

where $\mathbf{s}_1, \ldots, \mathbf{s}_n$ are samples from $S(\mathbf{s})$ and

$$\hat{\Phi}(\mathbf{x}) = \frac{1}{\sqrt{n}}\left[e^{-i\mathbf{s_1}\cdot\mathbf{x}}, \ldots, e^{-i\mathbf{s_n}\cdot\mathbf{x}}\right] \tag{5}$$

is the Fourier feature map approximating $k(\mathbf{x}, \mathbf{x}')$. In the case of the RBF kernel defined as

$$k(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{1}{2\sigma^2}\|\mathbf{x} - \mathbf{x}'\|_2^2\right), \tag{6}$$

[1] The *hat* in $\hat{\Phi}$ is used to indicate that the feature approximates a kernel in expectation.

where $\|\cdot\|$ is the Euclidean distance, the approximation is obtained in two steps: 1) we generate $n$ samples from $S(\mathbf{s}) \sim \mathcal{N}(0, 2\sigma^{-2}I)$ and $b \sim \text{uniform}\,[-\pi, \pi]$; 2) for each sample $i$ compute the feature approximation as $\cos(\mathbf{s}_i\mathbf{x} + b_i)$. The approximation is thus given by

$$\hat{\Phi}^{\text{Random}}(\mathbf{x}) = \frac{1}{\sqrt{n}}\left[\cos(\mathbf{s}_1\mathbf{x} + b_1), \ldots, \cos(\mathbf{s}_n\mathbf{x} + b_n)\right]. \tag{7}$$

In Eq. 7 we used the relation $e^{-i\mathbf{s}\cdot\mathbf{x}} = \cos(\mathbf{s}\cdot\mathbf{x}) - i\sin(\mathbf{s}\cdot\mathbf{x})$ and noted that the imaginary part must be zero for real kernels. Also note that $S(\mathbf{s})$ and $k(\tau)$ are duals and thus $S(\mathbf{s})$ can be obtained by calculating the inverse Fourier transform of $k(\tau)$. $b$ is symmetric about 0 and introduced to rotate the projection into the real axis by a random amount. This is known to produce better results in several practical problems [13].

### *B. Nyström features*

The Nyström method [22] approximates a kernel matrix $K$ by projecting it into a set of $m$ inducing points, denoted by $\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_m$. Then, $K \approx K_b\hat{K}^{\dagger}K_b^T$, where $K_b = [k(\mathbf{x}, \hat{\mathbf{x}})]_{N\times m}$ is a kernel matrix computed between all points in the dataset and the inducing points, $\hat{K} = [k(\hat{\mathbf{x}}_i, \hat{\mathbf{x}}_j)]_{m\times m}$ is a kernel matrix between the inducing points, and $\hat{K}^{\dagger}$ is the pseudo inverse of $\hat{K}$. Factorising the approximation into a feature vector yields:

$$\hat{\Phi}^{\text{Nyström}}(\mathbf{x}) = \hat{D}^{-1/2}\hat{V}^T\left(k(\mathbf{x}, \hat{\mathbf{x}}_1), \ldots, k(\mathbf{x}, \hat{\mathbf{x}}_m)\right)^T \tag{8}$$

where $\hat{D} = \text{diag}(\lambda_1, \ldots, \lambda_r)$ are the $r$ nonnegative eigenvalues of $\hat{K}$ in decreasing order and $\hat{V} = (\mathbf{v}, \ldots, \mathbf{v}_r)$ are the corresponding eigenvectors. It can be shown [21] that the Nyström approximation minimises the functional

$$\mathcal{E}(\hat{\Phi}) = \int \left(k(\mathbf{x}_i, \mathbf{x}_j) - \langle\hat{\Phi}(\mathbf{x}_i), \hat{\Phi}(\mathbf{x}_j)\rangle\right)^2 p(\mathbf{x}_i)p(\mathbf{x}_j)d\mathbf{x}_i\mathbf{x}_j \tag{9}$$

where $p(\mathbf{x}_i)$ and $p(\mathbf{x}_j)$ are approximated by a set of $r$ samples from the data. Therefore, the Nyström approximation is *nested*, i.e., it depends on the particular dataset being used. This is in contrast to *Random Fourier Features* which are dataset independent and can be computed *a priori*, once a specific kernel is defined.

### *C. Sparse random features*

The two approaches above can be used to approximate a RBF kernel but do not produce sparse features. With the goal to produce a sparse set of features that can be more easily optimised with SGD, we explore the properties of the sparse kernel introduced in [9]. The sparse kernel is defined as:

$$k_{sparse}(\mathbf{x}, \mathbf{x}') = \begin{cases} \left[\frac{2+\cos(2\pi r)}{3}(1-r) + \frac{1}{2\pi}\sin(2\pi r)\right] & \text{if} \quad r < 1 \\ 0 & \text{if} \quad r \geq 1 \end{cases} \tag{10}$$

where the matrix $\Omega$ is positive semi-definite and

$$r = \sqrt{(\mathbf{x} - \mathbf{x}')^T\,\Omega\,(\mathbf{x} - \mathbf{x}')}, \quad \Omega \geq 0. \tag{11}$$

This kernel has an important property that for distances $r \geq 1$ it returns 0. It also approximates the smoothness of a RBF kernel being four times differentiable. With this result, we define the sparse feature as,

$$\hat{\Phi}^{\text{Sparse}}(\mathbf{x}) = \left(k_{sparse}(\mathbf{x}, \hat{\mathbf{x}}_1), \ldots, k_{sparse}(\mathbf{x}, \hat{\mathbf{x}}_m)\right)^T, \tag{12}$$

where, as with the Nyström feature, $\hat{\mathbf{x}}_1, \ldots, \hat{\mathbf{x}}_m$ is a set of inducing points where the kernel is centred on. These inducing points can be uniformly sampled in the area the robot explores or can be placed in a grid.

### *D. Feature approximation of kernels on distributions*

In realistic mapping tasks there is typically uncertainty associated with the robot's position and imperfect sensor measurements. This uncertainty needs to be taken into account to accurately reflect the likelihood of occupancy of a given point in the map. We show how these uncertainties can be incorporated in Hilbert maps by deriving feature approximations to kernels over distributions.

Recent work in Kernel Embeddings has shown how to map probability distributions to a reproducing kernel Hilbert space (RKHS) [17, 19]. We follow this idea to derive our approximations. First, we assume that each point $\mathbf{x}$ is distributed as $\mathbb{P}$ in a probability space $\mathcal{P}$ in $(\mathcal{X}, \mathcal{A})$, where $\mathcal{X}$ is the input space and $\mathcal{A}$ is an associated $\sigma$-algebra. Let $\mathcal{H}$ denote a RKHS of functions $f : \mathcal{X} \to \mathbb{R}$ with a reproducing kernel $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$. The mean map $\mu$ from $\mathcal{P}$ into $\mathcal{H}$ can be obtained as,

$$\mu : \mathcal{P} \to \mathcal{H}, \quad \mathbb{P} \mapsto \int_{\mathcal{X}} k(x, \cdot)d\mathbb{P}(\mathbf{x}). \tag{13}$$

We can produce an empirical estimate of $\mu$ by drawing independent samples from $\mathbb{P}$ and creating a set $W = \{\mathbf{x}^{(1)} \ldots, \mathbf{x}^{(n)}\}$ such that

$$\hat{\mu}(\hat{\mathbb{P}}) = \frac{1}{n}\sum_{i=1}^{n} k(\mathbf{x}^{(i)}, \cdot). \tag{14}$$

This mean map estimator has been shown to converge to the mean map at a rate of $\mathcal{O}(n^{-\frac{1}{2}})$ in [17].

With the mean map estimator $\hat{\mu}$, a general positive semi-definite kernel $k(\mathbb{P}_i, \mathbb{P}_j)$ on distributions $\mathbb{P}_i$ and $\mathbb{P}_j$ can be approximated as follows:

$$\begin{aligned} k(\mathbb{P}_i, \mathbb{P}_j) &= \int\int \langle k(\mathbf{x}_i, \cdot), k(\mathbf{x}_j, \cdot)\rangle_{\mathcal{H}} d\mathbb{P}_i(\mathbf{x}_i)d\mathbb{P}_j(\mathbf{x}_j) && (15) \\ &= \int\int k(\mathbf{x}_i, \mathbf{x}_j)d\mathbb{P}_i(\mathbf{x}_i)d\mathbb{P}_j(\mathbf{x}_j) && (16) \\ &\approx \frac{1}{n}\frac{1}{m}\sum_{k=1}^{n}\sum_{l=1}^{m} k(\mathbf{x}_i^{(k)}, \mathbf{x}_j^{(l)}). && (17) \end{aligned}$$

In the above we used the reproducing property of $\mathcal{H}$ and the fact that $k(\mathbb{P}_i, \mathbb{P}_j) = \langle\mu_{\mathbb{P}_i}, \mu_{\mathbb{P}_j}\rangle_{\mathcal{H}}$. Finally, using the random Fourier, the Nyström or the sparse feature approximations as detailed above, the feature mapping approximation for a distribution $\mathbb{P}$ in $\mathcal{H}$ is

$$\hat{\Phi}(\mathbb{P}) = \frac{1}{n}\sum_{i=1}^{n}\hat{\Phi}(\mathbf{x}^{(i)}), \tag{18}$$

where $\mathbf{x}^{(i)}$ are samples in $W$ from $\mathbb{P}$.

## III. Online learning

The logistic regression model described in Section II can be learnt as part of an online optimisation procedure. However, in contrast to conventional logistic regression, the model operates on features $\hat{\Phi}(\mathbf{x})$ creating a nonlinear decision boundary.

### A. Objective function

To estimate the parameters $\mathbf{w}$ we minimise the regularised negative log-likelihood (NLL) given by:

$$NLL(\mathbf{w}) = \sum_{i=1}^{N} - \log p(y_i|\hat{\Phi}(\mathbf{x}_i), \mathbf{w}) + R(\mathbf{w}) \tag{19}$$

$$= \sum_{i=1}^{N} \log \Big(1 + \exp(-y_i \mathbf{w}^T \cdot \hat{\Phi}(\mathbf{x}_i))\Big) + R(\mathbf{w}), \tag{20}$$

where $R(\mathbf{w})$ is a regulariser to prevent overfitting and to enforce sparseness in $\mathbf{w}$. In this work we use the elastic net regulariser that has been shown to produce better results than L1 (LASSO) while preserving the same level of sparsity [26]. The elastic net regulariser is defined as,

$$R(\mathbf{w}) = \lambda_1 \|\mathbf{w}\|_2^2 + \lambda_2 \|\mathbf{w}\|_1, \tag{21}$$

where $\|\cdot\|_2$ and $\|\cdot\|_1$ are the L2 and L1 norms respectively, and $\lambda_1$ and $\lambda_2$ are parameters balancing the quadratic term (also called shrinkage parameter) and degree of sparseness, respectively.

The gradient of the objective function with respect to $\mathbf{w}$ can be computed as

$$\nabla NLL(\mathbf{w}) = \tag{22}$$

$$= \sum_{i=1}^{N} -y_i \hat{\Phi}(\mathbf{x}_i)(1 + \exp(y_i \mathbf{w}^T \cdot \hat{\Phi}(\mathbf{x}_i)))^{-1}$$

$$+ \frac{\partial R(\mathbf{w})}{\partial \mathbf{w}}. \tag{23}$$

Note that the $L_1$ term in $R(\mathbf{w})$ is non-differentiable so its derivative is generally approximated using sub-differentials.

### B. Stochastic gradient descent

One of the main advantages of utilising logistic regression is that the negative objective function in Eq. 20 can be optimised using fast stochastic gradient descent (SGD) methods. This is because the negative log-likelihood is the sum of the negative log-likelihoods of individual points. In contrast to batch algorithms such as Newton's method that require the computation of gradients and Hessians for all the points in the dataset SGD operates iteratively, giving a small step towards the goal with each data point.

To minimise Eq. 20, SGD iterates between randomly selecting a training point $\{\mathbf{x}_t, y_t\}$ from $\mathcal{D}$ and updating the parameters $\mathbf{w}$ as,

$$\mathbf{w}_t = \mathbf{w}_{t-1} - \eta_t A_t^{-1} \frac{\partial}{\partial \mathbf{w}} NLL(\mathbf{w}), \tag{24}$$

where $\eta > 0$ is known as the learning rate and matrix $A$ can be seen as a preconditioner to accelerate the convergence rate. In many cases, $A$ can be set to the identity matrix. This method is intrinsically online as new data points arriving from sensor measurements can be selected to update the parameters $\mathbf{w}$. Additionally, convergence analysis and generalisation behaviour have been extensively studied [25, 3]. It has been shown that even if SGD is applied to an unregularised version of Eq. 20, it achieves an implicit regularisation effect with good generalisation performance [1]. This facilitates the manual setting of the regularisation parameter as we know that even if we set it to zero, the model will still retain some resilience to overfitting.

The learning rate $\eta$ is either constant or asymptotically decaying with the number of iterations. In our implementation, we use the procedure proposed in [2] and set it to $\eta_t = \frac{1}{\lambda_1(t_0+t)}$, where $t_0$ is determined empirically from a small training set sampled from the full dataset.

Eq. 24 is effectively an online update procedure of the parameters. If the dataset grows, we can effectively select new points and update the parameters directly to reflect the new information. Conversely, we can shuffle the data, pass through each data point once, and repeat the process. This is known as the batch version of SGD [25]. Finally, we can average the parameters in the last $T$ iterations to remove some of the oscillation commonly seen during the optimisation. This is known as the averaged stochastic gradient descent [2, 23] and has been shown to improve on the convergence of conventional SGD for properly set learning rates [23]. Note that SGD has a constant cost per iteration and asymptotically converges to the expected risk [2].

## IV. Relationship to other methods

There are several classifiers in the machine learning literature that resemble the method described so far, each with pros and cons. Notably, Pegasos [16] is a different support vector machine (SVM) formulation where the expensive quadratic programming optimisation is replaced by stochastic gradient descent applied to the primal problem. The authors show that this method scales much better with the number of training points with strong convergence properties. Pegasos was not, however, trained on kernel feature approximations as our method and used a different loss function. The main reason we did not use a max-margin loss or hinge loss as with SVMs and chose the logistic regression formulation relates to the probabilistic interpretation of the results naturally obtained with logistic regression. An example can be seen in Fig. 1 where we show the same continuous occupancy map produced by the two methods. As SVMs do not produce a probabilistic interpretation directly, artificial probabilistic outputs are generally obtained using the method in [12]. However, as the figure shows, this introduces a problem; areas not explored, with no sensory information are classified as either occupied or non-occupied with high confidence. Conversely, with the logistic regression formulation, unexplored areas are classified with probability of 50% of being occupied as expected.

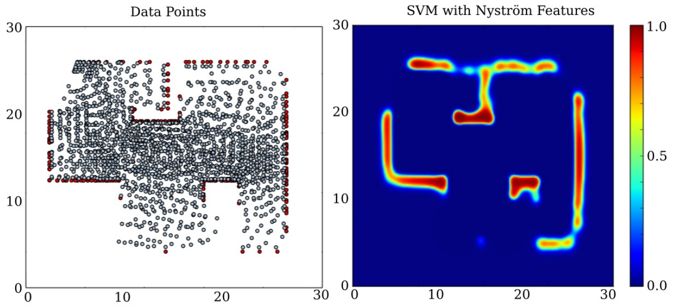

Fig. 1: Comparison between maps produced with Logistic Regression and SVMs. Left: Data points used for training, blue are non-occupied and brown are occupied. Right: Map produced with SVMs and Nyström features, with colour indicating probability of occupancy. The equivalent map generated with logistic regression is shown in Figure 6 (top left). Note that the SVM map is over confident about the occupancy status in areas with no data points, assumed unoccupied. This does not occur with the logistic regression map. Axis units are in metres.

This work also borrows ideas from Gaussian process occupancy maps (GPOMs) [11, 10] in that both attempt to represent the space in a continuous manner. Both produce probabilistic interpretations of occupancy, and both utilise kernels to represent data points in a high dimensional space. However, GPOMs suffer from a high computational cost and are significantly more difficult to be implemented. As a Bayesian method, GPOM do not have parameters that require manual tuning. That is an advantage compared to Hilbert maps. However, we noticed in our experiments that the regularisation parameters as well as the kernel parameters can be easily adjusted and the values do not need to be changed for different environments. Therefore, Hilbert maps offer significant computational advantages over GPOM at a reasonable price. Note also that Hilbert maps can be extended to a variational Bayesian logistic regression formulation as discussed in future work.

Another type of kernel approximation based on the Random Kitchen Sinks (RKSs) but with better scalability properties to high dimensional data was proposed in [7]. The authors show how to improve the cost of computing the features from $\mathcal{O}(nmd)$ to $\mathcal{O}(nm \log d)$, where $m$ is the number of features, $n$ is the number of samples and $d$ is the dimension of the inputs. The method is based on a fast factorised scheme to create the random matrices $\mathrm{s}$ in Eq. 7. Even though this extension is interesting for problems with high dimensional data such as in computer vision, it is less effective in our occupancy mapping problem as the dimensionality of the data is at most 3 for the $3D$ case. The implementation is also more complicated than the simple RKS so less likely to be appealing to real-time robotics applications.

An interesting study comparing RKS and Nyström features on several regression and classification problems was presented in [24]. The authors show that when there is a large gap between the eigen-spectrum $\hat{D}$ in Eq. 8, the Nyström method can produce impressive results and outperform RKS. As we shall see in the experiments, this was observed in the occupancy mapping problem where the Nyström method required a much smaller set of features to achieve similar accuracy. Note however, that the Nyström method is data dependent and the features cannot be precomputed as with RKS.

## V. Experiments

In the experiments unless stated otherwise we set the parameters of the model as follows: The kernel parameter $\sigma$ was 1.0, the number of components $m$ for each feature was Fourier=10k, Nyström=1k and sparse=2k. The regularisation parameters were $\lambda_1 = 0.0001$ and $\lambda_2 = 0.15$ (for the sparse case $\lambda_1 = 0.001$). These parameters were obtained through visual inspection of the results and remained unchanged for each of the maps we experimented with. Grid search can also be applied to set these parameters automatically.

### *A. Comparisons between the features*

In the first experiment we compare the three approaches to construct features for Hilbert maps. The experiment was conducted using the data from Intel-Lab (available at http://radish.sourceforge.net/). To better understand the generalisation power of the features, we created a series of occupancy maps where several beams from each observation were removed. The maps created were compared against test measurements retained for evaluation purposes and therefore not presented to algorithm. Figure 2 shows the maps created by the three features and the conventional occupancy grid maps for 10%, 30%, 60% and 100% of the original data incorporated. It can be seen that for both methods, the maps generated with Hilbert maps are much smoother and represent the uncertainty in the occupancy status of the environment more clearly. The occupancy grid maps, despite being sharper, contain a series of artefacts originated from anomalous observations generated from laser returns hitting non-reflective objects or influenced by glass. The Fourier features exhibit artefacts resulted from the cosine approximation to the RBF kernel in areas without observations, but perform comparably to the other two features in areas with more observations.

Figure 3 shows the area under the receiver operating characteristic (ROC) curve for the four approaches (Hilbert maps with Fourier, Nyström, sparse features, and occupancy grid maps) for several cases where a percentage of the observations is removed. It can be observed that the Sparse features and the Nyström perform the best. We can also observe that occupancy grid maps have problems in representing the environment properly when more than 50% of the data is removed. This indicates that Hilbert maps are more robust and possess better generalisation performance than occupancy grids.

In the second experiment we compare Hilbert maps with sparse features against occupancy grid maps for the outdoor dataset recorded at the University of Freiburg, also available at (http://radish.sourceforge.net/). Figure 4 shows the maps produced by both methods when 75% of the laser data is

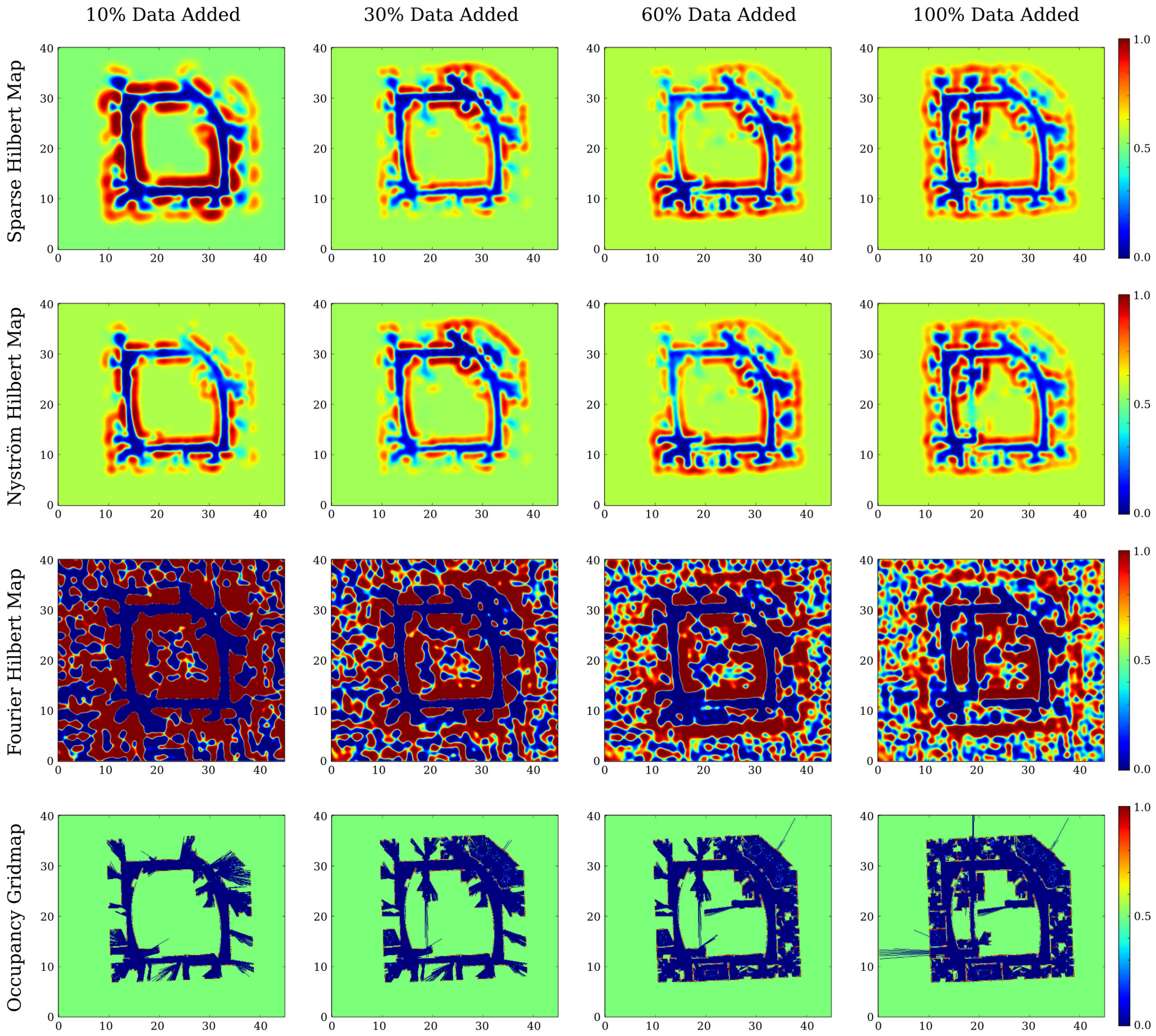

Fig. 2: Evolution of the different maps as observations are incrementally added. From top to bottom we have Sparse RBF, Nyström, Fourier and occupancy grid maps with 10%, 30%, 60% and 100% (left to right) of the data incorporated. Axis units are in metres.

removed from each scan. Removing data from the scans allows to better assess the generalisation power of both methods. For this test the regulariser parameter $\lambda_2$ was changed to 0.6 to exploit the sparsity of the data. It can be observed that the Hilbert map is significantly more resilient to outliers and noisy observations that naturally exist when navigating outdoors. Roads are more clearly identifiable and the map better reflects the actual shapes as can be seen in the aerial photo. Note that the width of the roads is exaggerated in the occupancy grids mostly due to spurious observations on vegetation. Quantitatively, this is confirmed in Table I where area under ROC for both methods is presented. Also note the overall uncertainty of the problem due to the noisy observations is much better handled. For this problem, due to the large number of observations, GPOM cannot be computed without sparse or nearest neighbour approximations that require storage of all the data.

| Method | Area under ROC curve | Runtime |
|---|---|---|
| Occupancy grid map | 0.61 | 88 s |
| Sparse Hilbert map | 0.80 | 850 s |

TABLE I: Area under the ROC curve and runtime for occupancy grid maps and sparse Hilbert maps when evaluated on the outdoor dataset with 75% of each laser scan missing.

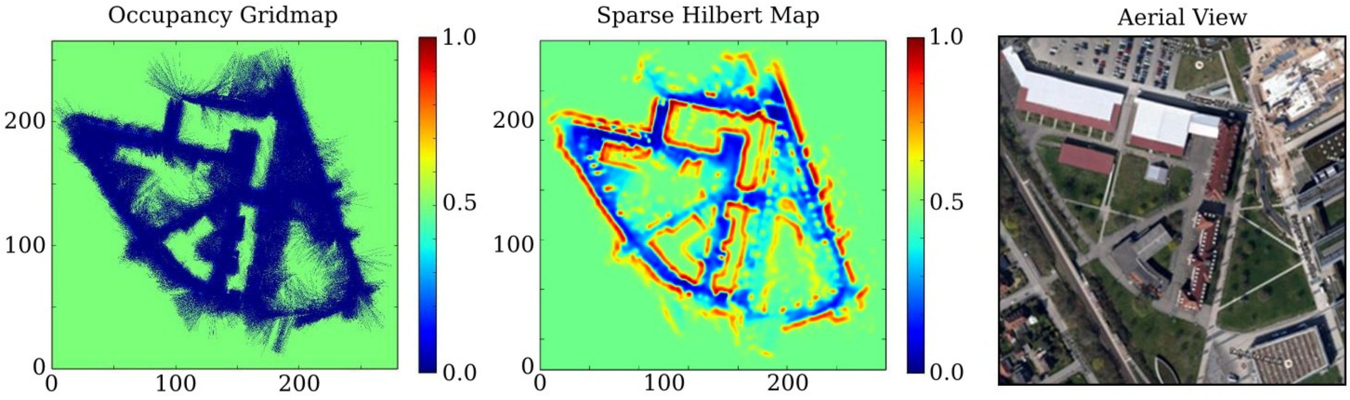

Fig. 4: Visualisation of the maps produced when 75% of the laser data is removed from each scan. The grid map (left) appears very noisy while the sparse Hilbert map (middle) shows obstacles and free areas clearly with areas barely observed by laser scans being marked as unknown. Comparing with the aerial image (right) we can see that the high certainty free areas correspond to footpaths along which the robot travelled. Axis units are in metres.

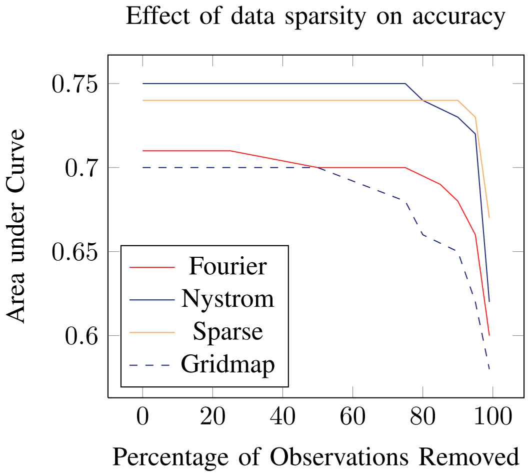

Fig. 3: Evolution of the area under curve for the three mapping methods when a variable amount of data is removed from each scan.

### B. Convergence of stochastic gradient descent

In this experiment we compare the convergence of SGD for the full batch case where all the data is presented to the algorithm multiple times and the incremental version where the algorithm sees each datapoint only once. This experiment was conducted on the Intel-Lab dataset with Nyström and sparse features. Figure 5 shows the value of the SGD objective being minimised with more iterations. For each iteration of the full SGD, all points are presented to the algorithm while for the incremental version, only a small set is presented without repetition. As expected, the incremental version oscillates more than the full version however both cases achieve a similar final energy value. The incremental version is significantly faster to execute; the full SGD takes approximately 21 seconds to complete 80 iterations with Nyström features and 1 second with sparse features while the incremental version takes 0.3 second with Nyström and 0.02 with sparse features. This result indicates the benefits of incremental SGD to quickly arrive at a solution even without observing the entire dataset. Also it shows that SGD can take advantage of sparse features to significantly speed up the computation.

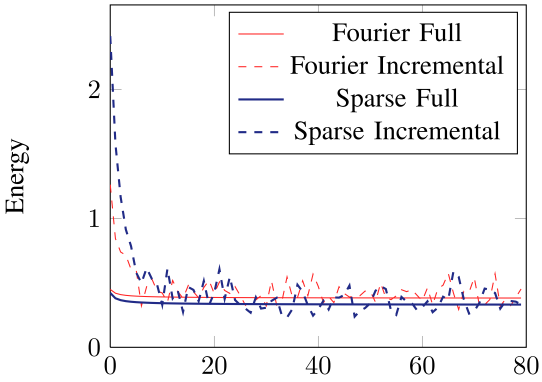

Fig. 5: Evaluation of the solution quality obtained when performing incremental SGD as opposed to full SGD with a fixed number of iterations using both Nyström and sparse features.

### C. Comparison with GPOM

Results comparing Hilbert maps with the three features against GPOM for the map in Figure 1 are presented in Table II. Both methods achieve remarkable classification results but as expected, GPOM are much more costly to compute, typically $\mathcal{O}(n^3)$ on the number of datapoints. Hilbert maps can produce similar results with an order of magnitude faster. Most of the computational cost in Hilbert maps is actually the feature computation rather than SGD. The Fourier features are the fastest followed by Sparse and Nyström that requires an eigen decomposition operation. Note however, that in both cases the features can be computed in parallel, thus presenting significant speedups on GPUs. The overall cost of Hilbert maps is $\mathcal{O}(m)$ on the number of features per datapoint.

| Method | Area under ROC curve | Runtime (s) |
|---|---|---|
| GPOM | 1.00 | 38.0 |
| Sparse Hilbert map | 1.00 | 3.2 |
| Fourier Hilbert map | 0.98 | 1.2 |
| Nyström Hilbert map | 0.98 | 6.9 |

TABLE II: Comparison of map quality and runtime of GPOM with the three proposed methods on the synthetic example shown in Figure 1. All methods outperform the GPOM by a large margin while obtaining identical or very similar results.

### D. Noisy inputs

In this experiment we demonstrate the ability of the features on distributions described in Section II-D to deal with errors in the position of observations. This also addresses the case where observations are partially observable and provided as distributions over the location. We use the synthetic dataset presented in Figure 1 but added noise to the position of the points. This simulates errors in localisation commonly present in real problems but allows us to compare against the ground truth. Figure 6 shows the original Hilbert map obtained when there is no noise in the data (top left). The data is then corrupted by Gaussian noise with 20 cm standard deviation (as a reference the size of the map is about 30 metres). The Hilbert map with Nyström features obtained on the corrupted dataset is displayed in the top right. It can be observed that walls and the shapes are much less defined. This is also the case for the occupancy grid maps (bottom right) where not even the walls can be properly identified. The Hilbert map result with Nytröm features on distributions generates the best result (bottom left) which resembles closely the result obtained when no noise is added to the data. This demonstrates the ability of the kernel approximation on distributions to handle highly noisy data.

## VI. Conclusion and future work

This paper introduced a novel occupancy mapping technique, the Hilbert maps. The techniques improves over occupancy grid maps in several ways but notably it does not require discretisation of the space providing maps at any resolution, and it captures spatial relationships to provide better generalisation in areas with no measurements while being more robust to outliers. The technique explores recent advancements in kernel machines, in particular kernel approximations, to allow efficient learning through stochastic gradient descent where strong convergence guarantees exist even when each data point is visited only once during learning. Experimental results were very encouraging showing that the maps produced are less influenced by outliers and more accurate in representing the underlying uncertainty.

An important advantage of online training strategies based on SGD relates to the speed in which maps can be updated after major trajectory corrections during navigation such as loop closures. Upon loop closure, SGD can be executed on the updated data until converge to create a new updated map. This procedure can be used in machine learning problems to deal with the problem of non-stationarity in sequential data (also known as covariance shift [20]). The advantage of Hilbert maps is that SGD is very fast and inexpensive to run therefore retraining can be performed efficiently.

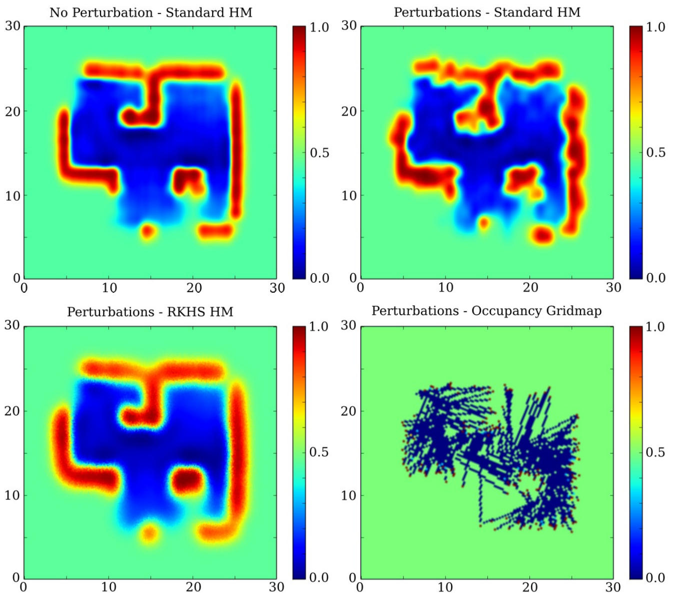

Fig. 6: Visualisation of the impact of noisy data. The top left image shows the result obtained when no perturbations are present using Hilbert maps. In the top right the data is perturbed with no compensation performed in the mapping. The bottom left image shows how the RKHS allows us to recover the map. The bottom right shows the effect data sparsity and noise has on occupancy grid maps which would not be suitable for actual use. Axis units are in metres.

There are several avenues for future work. First the paper did not explore the construction of 3D occupancy maps. Hilbert maps are general and should equally be able to represent 3D space; in fact some of its properties such as generalisation performance and ability to deal with data with different spatial density would be even more important in this case. Second, the method is formulated as a frequentist technique and requires the manual tuning of kernel and regularisation parameters. Even though recent techniques such as Bayesian optimisation can be used for this purpose [18], the more principled solution is to formulate the problem as Bayesian learning task. Unfortunately, this will lead to non-analytical solutions to the posterior and approximation techniques such as variational methods will need to be applied. Also, the objective function will no longer be convex and the SGD convergence guarantees in this case are less developed. Nevertheless this remains an interesting area for further investigation. Finally, occupancy mapping is just one example of many other problems where this technique can be applied. The fundamental idea of the algorithm which is to provide probabilistic predictions based on a stream of data captured by a moving robot in an online and efficient manner has a number of other applications. For example, the algorithm can be used to learn optimal policies in reinforcement learning, or to perform online object recognition.

# References

[1] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMSTAT*, pages 177–186, 2010.

[2] L. Bottou. Stochastic gradient descent tricks. In *Neural networks: Tricks of the trade*, pages 421–436, 2012.

[3] L. Bottou and O. Bousquet. The tradeoffs of large scale learning. In *Neural Information Processing Systems (NIPS)*, 2008.

[4] A. Elfes. Sonar-based real-world mapping and navigation. *IEEE Journal of Robotics and Automation*, RA-3(3):249–265, 1987.

[5] A. Elfes. *Occupancy grids: a probabilistic framework for robot perception and navigation*. PhD thesis, Carnegie Mellon University, 1989.

[6] I. I. Gihman and A. V. Skorohod. *The Theory of Stochastic Processes, volume 1*. Springer Verlag, Berlin, Germany, 1974.

[7] Q. Le, T. Sarlos, and A. Smola. Fastfood - Approximating kernel expansions in loglinear time. In *International Conference on Machine Learning (ICML)*, 2013.

[8] D. MacKay. Introduction to gaussian processes. In C. M. Bishop, editor, *Neural Networks and Machine Learning*, 1998.

[9] A. Melkumyan and F. Ramos. A sparse covariance function for exact gaussian process inference in large datasets. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1936–1942, 2009.

[10] S. T. O'Callaghan and F. T. Ramos. Gaussian process occupancy maps. *The International Journal of Robotics Research*, 31(1):42–62, 2012.

[11] S. T. O'Callaghan, F. T. Ramos, and H. Durrant-Whyte. Contextual occupancy maps using Gaussian processes. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3630–3636, 2009.

[12] J. C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, 1999.

[13] A. Rahimi and B. Recht. Random features for large-scale kernel machines. In *Neural Information Processing Systems (NIPS)*, 2008.

[14] A. Rahimi and B. Recht. Weighted sums of random kitchen sinks: Replacing minimization with randomisation in learning. In *Neural Information Processing Systems (NIPS)*, 2009.

[15] B. Scholkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001.

[16] Y. Singer and N. Srebro. Pegasos: Primal estimated sub-gradient solver for svm. In *International Conference on Machine Learning (ICML)*, pages 807–814, 2007.

[17] A. Smola, A. Gretton, L. Song, and B. Schlkopf. A hilbert space embedding for distributions. In *International Conference Algorithmic Learning Theory (COLT)*, pages 13–31. Springer-Verlag, 2007.

[18] J. Snoek, H. Larochelle, and R. P. Adams. Practical Bayesian optimization of machine learning algorithms. In *Neural Information Processing Systems (NIPS)*, 2012.

[19] L. Song, K. Fukumizu, and A. Gretton. Kernel embeddings of conditional distributions: A unified kernel framework for nonparametric inference in graphical models. *IEEE Signal Processing Magazine*, 30(4):98–111, 2013.

[20] M. Sugiyama, M. Krauledat, and K. Müller. Covariate shift adaptation by importance weighted cross validation. *Journal of Machine Learning Research*, 8:985–1005, December 2007.

[21] C. Williams and M. Seeger. The effect of the input density distribution on kernel-based classifiers. In *International Conference on Machine Learning (ICML)*, pages 1159–1166. Morgan Kaufmann, 2000.

[22] C. K. I. Williams and M. Seeger. Using the Nyström method to speed up kernel machines. In *Neural Information Processing Systems (NIPS)*, 2000.

[23] W. Xu. Towards optimal one pass large scale learning with averaged stochastic gradient descent. *CoRR*, abs/1107.2490, 2011.

[24] T. Yang, Y. Li, M. Mahdavi, R. Jin, and Z. Zhou. Nyström method vs random fourier features: A theoretical and empirical comparison. In *Neural Information Processing Systems (NIPS)*, pages 476–484, 2012.

[25] T. Zhang. Solving large-scale linear prediction problems using stochastic gradient descent algorithms. In *International Conference on Machine Learning (ICML)*, 2004.

[26] H. Zou and T. Hastie. Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society, Series B*, 67:301–320, 2005.