Bayesian Hilbert Maps for Dynamic Continuous Occupancy Mapping

Ransalu Senanayake
School of Information Technologies
The University of Sydney
Australia
rsen4557@uni.sydney.edu.au

Fabio Ramos
School of Information Technologies
The University of Sydney
Australia
fabio.ramos@sydney.edu.au

Abstract: Hilbert mapping is an efficient technique for building continuous occupancy maps from depth sensors such as LiDAR in static environments. However, to make the map adaptable to dynamic environments, its parameters need to be learned automatically. In this paper, we take a variational Bayesian approach to this problem, thus eliminating the regularization term typically adjusted heuristically. We extend the proposed model to learn long-term occupancy maps in dynamic environments in a sequential fashion, demonstrating the power of kernel methods to capture abstract nonlinear patterns and Bayesian learning to construct sophisticated models. Experiments conducted in environments with moving vehicles show that the proposed approach has a significant speed improvement over the state-of-the-art techniques and maintain a similar or better accuracy. We also discuss the robustness against occlusions and various theoretical and empirical aspects of building long-term dynamic occupancy maps.

Keywords: Robots, RKHS, Spatiotemporal, Dynamic environments, SLAM

1 Introduction

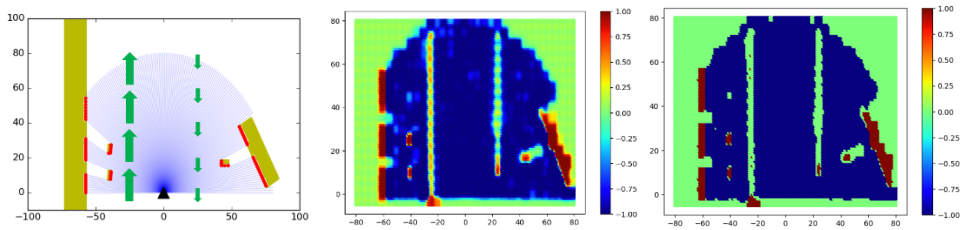
Distinguishing occupied areas from unoccupied areas in unseen and unstructured environments is central to path planning in autonomous vehicles. This task becomes even more challenging in the presence of dynamic objects such as moving vehicles. Almost all fully autonomous vehicles — commercial driverless cars such as Uber, Google, etc. and trucks used in the mining industry [1]— are equipped with depth sensors such as LiDAR as they require to build occupancy maps from *sparse* range measurements.

Typically, occupancy grid maps have been used for modeling the occupancy state of the environment by dividing the world into a fixed-sized grid and then applying a Bayes filter to cells individually [2]. In order to alleviate its limitations such as, 1) the requirement of predefining a cell size, 2) strong assumptions of independence between nearby cells, Gaussian process occupancy maps (GPOMs) [3, 4] have been developed. The covariance function of the Gaussian process naturally considers neighborhood information making GPOMs robust against occlusions.

As opposed to occupancy grid maps, GPOMs model the occupancy as a function which can be easily queried to evaluate the occupancy probability of any point in the environment. Such probabilistic frameworks can be effectively used for path planning with safety in mind [5, 6], and perform simultaneous mapping and planing [7] under one framework thanks to kernelization [8]. Nevertheless, being a non-parametric model, the computational complexity of GPOM is $\mathcal{O}(N^3)$ for N data points which unmanageably grows as more laser scans are obtained over time. Although [9] uses stochastic variational inference to make GPOMs faster, the computational time still increases with N.

Bringing all advantages of GPOMs, [10] proposed Hilbert maps (HMs)—a parametric model based on another kernel method. It takes the form of a kernelized logistic regression classifier which attempts to minimize the regularized negative log-likelihood using stochastic gradient descent to estimate its parameters. Although it has been extended for static 3D environments [11, 12] as well

1st Conference on Robot Learning (CoRL 2017), Mountain View, United States.



(a) Motion field (b) BHM or SBHM (c) The three states

Figure 1: The long-term occupancy map produced using the proposed algorithm after observing the field for several minutes. The robot indicated by the black arrow head resides at the middle of the two roads. Its field of view is shown in blue laser beams with red laser hit points, when there are no moving vehicles. Static objects such as buildings and parked vehicles are shown in yellow and the traffic flow in green arrows. Vehicles moving in the upward direction are more frequent than that of downward. Therefore, after several laser observations, the occupancy probability of the left road shown in (b) is higher than that of the right road. Always occupied areas are indicated by $+1$ while unoccupied areas are indicated by -1. The occupancy level of the unseen outlying areas is almost 0, i.e. uncertain. (c) is the tristate BHM map showing occupied, unoccupied, and undecidable was obtained by rounding the occupancy levels to the nearest integers$\in \{-1, 0, +1\}$. Since the model captures neighborhood relationships, areas around $(-50, 15)$, $(-50, 35)$ and $(60, 20)$ are correctly predicted, regardless of occlusions due to the three parked vehicles.

as to predict short-term dynamics [13], it cannot be directly used for mapping long-term dynamics—areas which are generally occupied when observed over a period of time, as in the model illustrated by the example in Figure 1. This is because it internally uses a Gaussian process regression model with a squared-exponential kernel in the time domain.

As another limitation of Hilbert maps, its objective function contains a regularization term which needs to be pre-fixed. The regularization parameters are used to prevent over-fitting and maintain a consistent sparsity in the map, especially in areas where no data are available. This will be further discussed in Section 2.3. Making a multitude of significant improvements to make static Hilbert maps readily usable in complex environments, this paper presents the following contributions;

1. a variational Bayesian formulation to Hilbert maps, eliminating crucial regularization parameter tuning;
2. a sequential learning framework that does not store observed data, and;
3. a long-term spatiotemporal model with an almost fixed time-memory budget.

To the best of our knowledge, previous methods that concern about dynamics of the environment when building occupancy maps are either formulated for extracting deterministic patterns [14, 15, 16, 17, 18] or eliminating dynamic objects from the environment in order to build robust static maps [19, 20]. In contrast, as in [9], our method focuses on developing long-term dynamic maps which can later be used for path planning. However, unlike [9], our method does not require previous data making the learning process significantly faster as well as being memory frugal.

We start the following section by introducing Hilbert maps, and then, in Section 3, we propose its "fully Bayesian" treatment (named BHM) which requires data to be accumulated over time to robustly build a long-term model. Then, in Section 4 we run it sequentially (named SBHM) without requiring old data to update the model. Experiments and discussions are given in Section 6.

2 Hilbert Maps

The Hilbert maps framework [10] is developed for building continuous occupancy maps in static environments. It makes use of regularized logistic regression to model occupied and unoccupied states, and optimizes its model parameters using stochastic gradient descent (SGD).

2.1 Data

It is assumed that data points are collected from a line-of-sight depth sensor such as LiDAR or sonar. The end point of each beam, when it hits an obstacle, is labeled as an occupied point $y = 1$, and

samples drawn from a uniform distribution with a support between the sensor and the end point are labeled as unoccupied points $y = -1$. The spatial locations, latitude and longitude, corresponding to each y are denoted by $\mathbf{x} \in \mathbb{R}^2$. N such input-output pairs $\{\mathbf{x}_n, y_n\}_{n=1}^N$ will be used for supervised learning.

2.2 The Hilbert maps model

Hilbert maps are based on an approximate kernel defined by the inner product $\text{kern}(\mathbf{x}, \tilde{\mathbf{x}}) \approx \Phi(\mathbf{x})^\top \Phi(\tilde{\mathbf{x}})$ with features $\Phi(\cdot)$. Although three different features are suggested in [10], our discussion will be based on hinged features defined by,

$$k(\mathbf{x}, \tilde{\mathbf{x}}) = \exp\big(-\gamma \|\mathbf{x} - \tilde{\mathbf{x}}\|^2\big), \tag{1}$$

as they have a physical meaning and, as also concluded by authors, they experimentally outperform other features. Here, γ is the bandwidth which controls the width of the Gaussian-shaped curve, and $\tilde{\mathbf{x}}$ is a spatially fixed point in the environment. Having D such points (the higher, the richer) hinged in different locations of the environment, the feature vector can be computed by,

$$\Phi(\mathbf{x}) = \big(1, k(\mathbf{x}, \tilde{\mathbf{x}}_1), k(\mathbf{x}, \tilde{\mathbf{x}}_2), ..., k(\mathbf{x}, \tilde{\mathbf{x}}_D)\big), \tag{2}$$

The probability that a point in the environment is not-occupied is defined by the sigmoid function,

$$\mathrm{P}(y = -1|\mathbf{x}, \mathbf{w}) = \Big(1 + \exp\big(\mathbf{w}^\top \Phi(\mathbf{x})\big)\Big)^{-1} =: \sigma\big(-\mathbf{w}^\top \Phi(\mathbf{x})\big). \tag{3}$$

The parameters $\mathbf{w}$ are learned by minimizing the regularized negative log-likelihood,

$$\mathrm{RNLL} = \sum_{i=1}^{N} \log\Big(1 + \exp\big(-y_i \mathbf{w}^\top \Phi(\mathbf{x})\big)\Big) + \underbrace{\Big(\alpha_1 \|\mathbf{w}\|_1 + \alpha_2 \|\mathbf{w}\|_2^2\Big)}_{Regularization}, \tag{4}$$

with α_1 and α_2 regularization parameters.

2.3 Importance of regularization

The regularization term in (4) is commonly known as the elastic-net regularizer which can be thought as a convex combination of Lasso and Ridge regularization. The L1 norm controls the sparsity while the L2 norm controls the convexity. As illustrated in Figure 2, the Hilbert maps model heavily depends on the regularization and requires careful tuning.

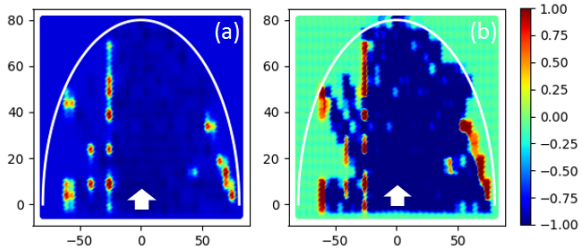


Figure 2: The white arrows show the position of the robot while the white arcs show the range the robot can see. (a) Hilbert maps "without regularization" predict areas where there are no data as unoccupied—majority dark blue, (b) Bayesian Hilbert maps which does not require a regularization term intrinsically identify such areas as unknown—green.

3 Bayesian Hilbert Maps (BHMs)

3.1 The model

As with HMs, data are collected as discussed in Section 2.1 and features $\Phi(\mathbf{x})$ are computed using (2). However, unlike HMs, similar to our workshop paper [21], we will take a Bayesian approach, effectively eliminating the requirement of regularization terms, rather than minimizing the regularized negative log-likelihood. As an added advantage of Bayesian learning, the proposed approach only requires a little amount of data to learn the large parameter vector $\mathbf{w}$. However, it is not possible to obtain an analytical solution for the posterior of the Bayesian model because of the sigmoid

likelihood, and hence, as indicated by (5), the posterior is approximated by another distribution Q with parameters $\mathbf{w}$ and α. Each of its terms will be elaborated in the following sections.

$$\underbrace{\mathrm{Q}(\mathbf{w},\alpha)}_{\substack{\text{approx.}\\\text{posterior}}} \approx \underbrace{\mathrm{P}(\mathbf{w},\alpha|\mathbf{x},\mathbf{y})}_{\text{posterior}} = \frac{\overbrace{\mathrm{P}(\mathbf{y}|\mathbf{x},\mathbf{w})}^{\text{likelihood}} \times \overbrace{\mathrm{P}(\mathbf{w}|\alpha)}^{\text{prior}} \times \overbrace{\mathrm{P}(\alpha)}^{\text{hyper-prior}}}{\underbrace{\mathrm{P}(\mathbf{y})}_{\text{marginal likelihood}}} \tag{5}$$

3.1.1 The likelihood

As discussed in Section 2.1, LiDAR data points are independent from each other and hence the data likelihood can be written as, $\mathrm{P}(\mathbf{y}|\mathbf{x},\mathbf{w}) = \prod_{n=1}^{N} \mathrm{P}(y_n|\mathbf{x}_n,\mathbf{w}) = \prod_{n=1}^{N} \sigma\big(y_n \mathbf{w}^\top \Phi(\mathbf{x}_n)\big)$. This sigmoid likelihood does not have a conjugate prior. Therefore, it will be locally approximated by the exponential of a quadratic form in such a way a standard prior distribution can be used to make the posterior evaluable using variational inference [22].

Theorem 1 *[23] A sigmoid likelihood $\sigma(yr) := \mathrm{P}(y|r)$ can be lower bounded by,*

$$\sigma(yr) \geq \sigma(\xi) \exp\Big[\frac{\xi - r}{2} - \frac{1}{2\xi}\Big(\sigma(\xi) - \frac{1}{2}\Big)\big(r^2 - \xi^2\big)\Big], \tag{6}$$

with ξ as a local parameter used to linearize the function using the Taylor expansion.

Letting $r = \mathbf{w}^\top \Phi(\mathbf{x})$ in Theorem 1 provides a lower bound for the data likelihood defined in (3.1.1). ξ is a parameter that needs to be learned from data.

3.1.2 The prior and posterior distributions

Rather than having a pre-defined hyperparameter $\boldsymbol{\alpha}$ in the prior $\mathrm{P}(\mathbf{w}|\boldsymbol{\alpha})$, the objective is to learn it from data itself. The prior is a Gaussian $\mathrm{P}(\mathbf{w}|\boldsymbol{\alpha}) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \Sigma_\alpha)$ with Σ_α being a diagonal matrix with diagonal elements α_d^{-1} created from a hyperparameter vector $\boldsymbol{\alpha}$. In order to learn $\boldsymbol{\alpha}$, a Gamma hyper-prior $\mathrm{P}(\boldsymbol{\alpha}) = \prod_{d=1}^{D} \Gamma(\alpha_d|a_0, b_0)$ is used, where constants a_0, b_0 are shape parameter and scale parameter, respectively.

Considering the mean-field approximation [22], the posterior distribution is factorized as $\mathrm{Q}(\mathbf{w},\alpha) = \mathrm{Q}(\mathbf{w})\mathrm{Q}(\boldsymbol{\alpha})$, where $\mathrm{Q}(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $\mathrm{Q}(\boldsymbol{\alpha}) = \Gamma(\boldsymbol{\alpha}|a, \mathbf{b}) = \prod_{d=1}^{D} \Gamma(\alpha_d|a, b_d)$. In the learning phase, it is required to learn parameters $\boldsymbol{\mu}, \boldsymbol{\Sigma}, a$, and $\mathbf{b}$ to accurately model occupancy states.

3.2 Learning parameters

The marginal likelihood, $\mathrm{P}(\mathbf{y}|\mathbf{x}) = \int\int \mathrm{P}(\mathbf{y}|\mathbf{x},\mathbf{w})\mathrm{P}(\mathbf{w}|\alpha)\mathrm{P}(\alpha)\mathrm{d}\mathbf{w}\mathrm{d}\alpha$ is intractable, and hence, in variational inference, the log-marginal likelihood is decomposed as,

$$\ln \underbrace{\mathrm{P}(\mathbf{y}|\mathbf{x})}_{\substack{\text{marginal}\\\text{likelihood}}} = \mathcal{L}\big(\underbrace{\mathrm{Q}(\mathbf{w},\alpha)}_{\substack{\text{approx.}\\\text{posterior}}}\big) + \mathbb{KL}\big(\underbrace{\mathrm{Q}(\mathbf{w},\alpha)}_{\substack{\text{approx.}\\\text{posterior}}} \| \underbrace{\mathrm{P}(\mathbf{w},\alpha|\mathbf{x},\mathbf{y})}_{\text{posterior}}\big), \tag{7}$$

where,

$$\mathcal{L} = \int\int \mathrm{Q}(\mathbf{w},\alpha) \ln\left(\frac{\mathrm{P}(\mathbf{w},\alpha,\mathbf{y})}{\mathrm{Q}(\mathbf{w},\alpha)}\right)\mathrm{d}\mathbf{w}\mathrm{d}\alpha, \text{ and } \mathbb{KL} = -\int\int \mathrm{Q}(\mathbf{w},\alpha) \ln\left(\frac{\mathrm{P}(\mathbf{w},\alpha|\mathbf{y})}{\mathrm{Q}(\mathbf{w},\alpha)}\right)\mathrm{d}\mathbf{w}\mathrm{d}\alpha, \tag{8}$$

are the lower bound and Kullback-Leibler (KL) divergence, respectively. Compared to Markov chain Monte Carlo (MCMC) techniques, variational inference have shown to be well suitable for learning high dimensional models with large datasets within a short period of time [24, 25]. Typically, in variational inference, the objective is to find $\mathbf{w}$ and α that minimize the distance, i.e. KL-divergence, between the approximate posterior and true posterior. However, since computing KL-divergence requires access to the true posterior which we do not have, instead of minimizing the KL term,

the lower bound $\mathcal{L}$ is maximized, considering the fact that marginal likelihood does not depend on parameters.

Unlike in the generic setting, $\mathcal{L}$ is also not explicitly computable here because of the sigmoid likelihood. Therefore, combining the bound (6) with the decomposition (7), a new lower bound $\tilde{\mathcal{L}}(\mathrm{Q}, \xi) \leq \mathcal{L}(\mathrm{Q})$ is obtained. The variational parameters can be learned iteratively as an Expectation-Maximization (EM) procedure. As shown in Algorithm 1, in each iteration, ξ values are fixed to update $\mu, \mathbf{\Sigma}, a$, and b in the E-step, and vice versa in the M-step. *concatenate*() is the data accumulation procedure while *learn_*() are parameter estimation functions. All relevant equations are given in the supplementary materials. *filter*() function can be thought as a filtering procedure to select the most informative points, and it will be discussed in Section 5.

The bandwidth parameter may be approximately learned by maximizing $\mathcal{L}$ w.r.t. γ using gradient descent in the inner *while*-loops of the algorithms. Although the more theoretically sound approach would be to define a distribution over γ and make further approximations, as also highlighted in experiments, such procedures are intractable and redundant for this particular robotics application.

$t \leftarrow -1$
$a_0, b_0 \leftarrow$ small values
while *sensor is active* **do**
  $t \leftarrow t + 1$
  Get new scan $\mathcal{D}_t = \{(\mathbf{x}_t, y_t)\}$
  if $t = 0$ **then**
    $\mathcal{D} \leftarrow \mathcal{D}_t$
  else
    $\mathcal{D}_t \leftarrow filter(\mathcal{D}_t)$
    $\mathcal{D} \leftarrow concatenate(\mathcal{D}, \mathcal{D}_t)$
  end
  $\xi = 0$
  while *not converged* **do**
    $\mu, \Sigma \leftarrow learn\_\mathbf{w}(\xi, \mathcal{D})$
    $a, b \leftarrow learn\_\alpha(\xi, a_0, b_0, \mathcal{D})$ } E-step
    $\xi \leftarrow learn\_local(\mu, \Sigma, a, b, \mathcal{D})$ } M-step
  end
end

Algorithm 1: BHM

$t \leftarrow -1$
$\sigma_0 \leftarrow$ a small value
while *sensor is active* **do**
  $t \leftarrow t + 1$
  Get new scan $\mathcal{D}_t = \{(\mathbf{x}_t, y_t)\}$
  if $t = 0$ **then**
    $\mathcal{D} \leftarrow \mathcal{D}_t$
    $\mu_0, \Sigma_0 \leftarrow \mathbf{0}, \sigma_0^{-1}\mathbf{I}$
  else
    $\mathcal{D} \leftarrow filter(\mathcal{D}_t)$
    $\mu_t, \Sigma_t \leftarrow \mu_{t-1}, \Sigma_{t-1}$
  end
  $\xi = 0$
  while *not converged* **do**
    $\mu_t, \Sigma_t \leftarrow learn\_\mathbf{w}(\xi, \mu_t, \Sigma_t, \mathcal{D})$ } E-step
    $\xi \leftarrow learn\_local(\mu_t, \Sigma_t, \mathcal{D})$ } M-step
  end
end

Algorithm 2: SBHM

4 Sequential Bayesian Hilbert Maps (SBHMs)

In Section 3 we described the fully Bayesian treatment to Hilbert maps where hypeparameters α are also learned by the model itself. Even though the model is theoretically appealing, it has several shortcomings from an application point of view. Observe that the parameter structure of the prior distributions is different to that of the posterior and hence the posterior in step $t-1$ cannot be directly feedbacked as the prior at step t. Nevertheless, as shown in supplementary materials, since the prior and posterior of BHM are interconnected merely by the expectation of all hyperparameters $\mathbb{E}(\alpha)$, the estimation of $\mathrm{Q}(\mathbf{w})$ is biased towards the data at time t, if all past data are discarded. This is why BHM requires data to be accumulated over time to build an unbiased and robust long-term map.

To ameliorate the issues, we redefine the BHM prior as $\mathrm{P}(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mu_{t-1}, \mathbf{\Sigma}_{t-1})$ and the posterior as $\mathrm{Q}(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mu_t, \mathbf{\Sigma}_t)$. Observe that, not only they share the same distribution structure—Gaussian distributions with non-zero means and covariances—but also the posterior estimation of the previous time step is used as the prior of the current time step. The assumption that the posterior estimate of the previous step is good enough—captures all useful information—to be used as a prior for the current step is validated in experiments, and the philosophy is illustrated in supplementary materials following a simple Bayesian linear regression example given in [26]. Intuitively, simple multivariate Gaussian distributions are sufficient for this application because we work in a rich high dimensional feature space which is capable of absorbing non-linear spatial variations.

At $t = 0$, we start to learn with a diffuse prior $\mathcal{N}(\mu_0, \mathbf{\Sigma}_0) = \mathcal{N}(\mathbf{0}, \sigma_0^{-1}\mathbf{I})$ where $\sigma_0 \approx 0$, and we experimentally verify that a good accuracy is obtained without having a hyperprior. Comparing

Algorithms 1 and 2, note that BHM not only requires data to be aggregated over time but also it requires two posterior distributions to be computed. In contrast, in SBHMs, not only all previous data are discarded but also posterior estimations are straightforward as the model is much simpler, making overall computations significantly faster.

5 Information filtering

In both methods, for each data point in each new sequential scan $\mathbf{x}_*$, except at $t = 0$, occupancy level $f(\mathbf{x}_*)$ is computed. If each data point satisfies the criterion $|f(\mathbf{x}_*) - y_{\text{true}}| \geq \eta$ such data points are used for learning the map. Here, $f(\cdot)$ is the function that is used to query from the map that has been learned before incorporating the data from the new scan, and y_{true} are the actual occupancy state $\in \{+1, -1\}$ of each point in the current scan. As illustrated in Figure 3, this filters points that can provide new information for an arbitrary threshold η. For instance, if a vehicle is moving into a new area, new information around that area will be higher than that of stationary areas. Although we attempted to use cross entropy, the aforementioned criterion experimentally provided better results.

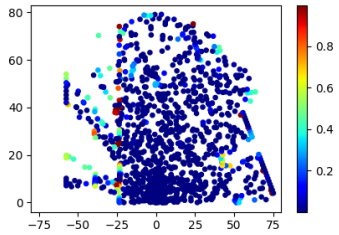


Figure 3: Having learned the map for $t-1$ time steps, the information gain is calculated for each point in the t^{th} scan. The higher values, i.e. red, indicate new information such as an area a vehicle has entered. Therefore, adding data points with smaller values, say values < 0.3, does hardly improve the accuracy.

6 Experiments

6.1 The experimental setup and evaluation

Three datasets, taken from [9], will be used for experiments. Dataset 1 is obtained from a simulator which resembles a 80 m LiDAR covering 180^0. As shown in Figure (1), this dataset has been designed in such a way that vehicles travel more often in the left lane than in the right lane. The walls in the sides of the roads are partially occluded by the parked vehicles. Datasets 2 is taken from a real world four-way intersection with vehicles turning in different directions following traffic lights. Its LiDAR covers 180^0 in a 30 m radius. Dataset 3 has also been captured from a real road near a four-way intersection using a 60 m LiDAR which can see 270^0. The code is available at https://github.com/RansML/Bayesian_Hilbert_Maps.

In order to evaluate the occupancy level $\mathrm{P}(y = 1|\mathbf{x}_*, \mathbf{x}, \mathbf{y})$ of any point in the environment $\mathbf{x}_*$ using the proposed approaches (BHM and SBHM), the posterior distribution is marginalized. Our models will be compared against, 1) variational sparse dynamic Gaussian process occupancy maps (VSDGPOM) [9] which is capable of building similar maps to our approaches, 2) dynamic Gaussian process occupancy maps (DGPOM) [27], and 3) dynamic Grid maps (DGrid), an extension of occupancy grid maps to dynamic environments by keeping memory in each cell individually [28].

All experiments were run on a laptop with 8 GB RAM. It was assumed that the robot is stationary and the localization is given. As in [9], the area under the receiver operating characteristic (ROC) curve (AUC) will be used as the fundamental measure for comparisons. As an additional metric, negative log-likelihood loss (NLL), also known as cross entropy, defined by $-\log p(y|y_*) = -y \log(y_*) + (1-y)\log(1-y_*)$ will be used for evaluating accuracy in the spatiotemporal setting. Although the model is hardly sensitive to its initial parameter settings, what we have used are given in the supplementary materials. Filtering threshold η is the only parameter that needs to be chosen by the user to maintain a desired speed-accuracy trade-off and it has been fixed to 0.3 throughout all experiments.

6.2 The effect of the bandwidth parameter

Firstly, we visualize how the bandwidth parameter of the kernel affects the smoothness of the map. As shown in Figure 4, large γ values tend to produce less smooth maps while capturing sharp edges.

In contrast, small γ values produce smoother maps because each point has a very high influence on even farther neighbors making an average over a relatively large realm.

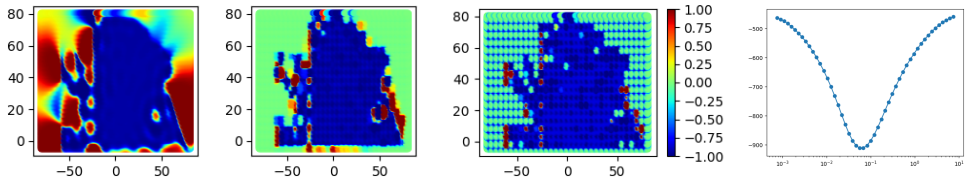


Figure 4: The effect of γ for the LiDAR scan at $t = 0$ in dataset 1. (a)-(c) $\gamma = (0.0075, 0.025, 0.75)$ (d) A sample plot of the negative loss function w.r.t. γ

6.3 Spatial accuracy

To determine how well the occupancy probability of a given location can be predicted accurately, always occupied areas such as walls, parked vehicles, and always unoccupied areas were labeled manually. Note that this test dataset contains both occluded and non-occluded areas of the environment. The AUC is calculated over a period of time as the map is built sequentially. The last three numerical columns of Table 1 indicate that spatial accuracy of BHMs and SBHMs is comparable or better than Gaussian process based techniques, while significantly higher than grid maps.

The first column of Table 1 reports the accuracy of predicting occupancy state in occluded areas such as behind parked vehicles which can be only labeled for the simulation dataset. As expected, unlike the other three kernel methods, the grid based method is not robust against occlusions [3] as it does not consider neighborhood information. However, on a different note, if the occlusions are large and the area is not visible at all, obviously, even the kernel based methods will not be robust against occlusions, and the accuracy will not be close to one. Nevertheless, in comparison with DGrid, here we demonstrate the advantage of considering neighborhood information.

Table 1: Average AUC ($\mu \pm 2\sigma$) for labeled spatial data. The last three columns show accuracy based on randomly selected points from the environment and the first column indicates accuracy of predicting occluded areas.

Method	Dataset 1 Occlu.	Dataset 1	Dataset 2	Dataset 3
SBHM	1.00 ± 0.01	0.99 ± 0.04	1.00 ± 0.00	0.96 ± 0.05
BHM	1.00 ± 0.00	1.00 ± 0.01	1.00 ± 0.02	0.95 ± 0.01
VSDGPOM	1.00 ± 0.00	0.99 ± 0.04	1.00 ± 0.00	0.94 ± 0.05
DGPOM	0.99 ± 0.02	0.99 ± 0.02	0.98 ± 0.08	0.96 ± 0.02
DGrid	0.50 ± 0.00	0.78 ± 0.04	0.84 ± 0.17	0.91 ± 0.02

6.4 The effect of information filtering for dynamic mapping

As discussed in Section 5, because of the sampling procedure it is possible to obtain finitely many data points for a single scan, and hence it is vital to filter informative data. Figure 5 illustrates speed-accuracy trade-off for different threshold η values. The performance metrics are evaluated for each step as the model is learned sequentially. The test dataset contains past, present, and future data which are never used for training purposes.

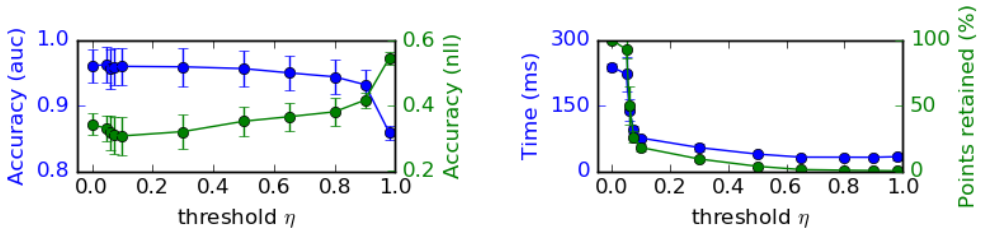


Figure 5: Speed-accuracy trade-off determined by η for dataset 1. Setting $\eta = 0.1$ filters $\approx 80\%$ of uninformative data and making the algorithm at least 3 times faster without deteriorating the accuracy.

6.5 Learning long-term maps

In this experiment, we demonstrate building spatiotemporal maps by following the experimental procedure in Section 6.4. Figures 1 and 6 illustrate such long-term occupancy maps—which areas of the environment are occupied in general—for the three datasets.

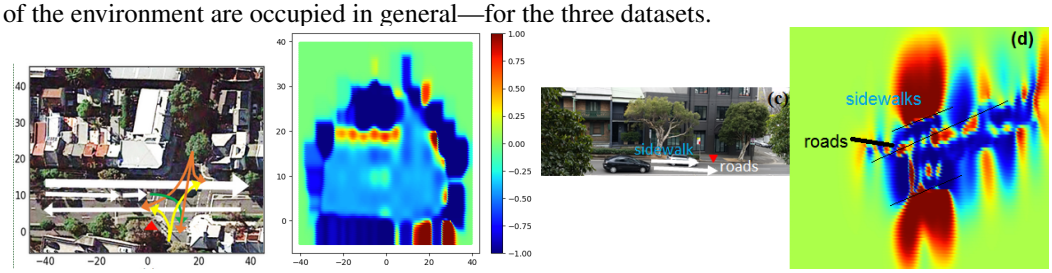


Figure 6: The satellite map of datasets 2 and 3 and their corresponding SBHMs. The arrows indicate the traffic flow of the busy four-way intersections.

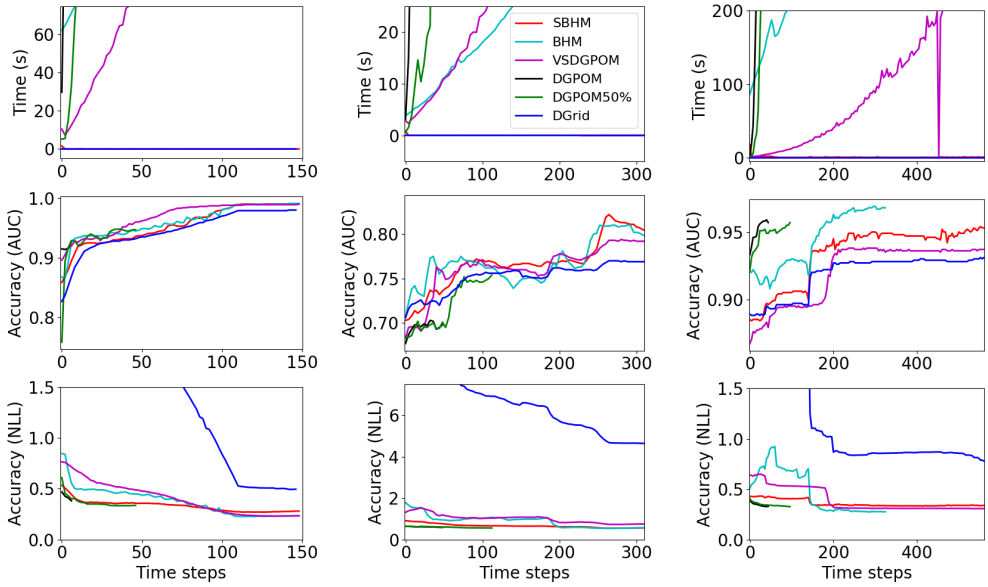


Figure 7: Left to right columns: Datasets 1, 2, and 3. The first row shows the time performance while the last two rows show accuracy metrics. SBHMs have an almost constant cost and significantly faster than the other competing methods to obtain a better or similar accuracy. DGPOM50% is DGPOM with 50% of data. Note: 1) SBHM (red) and DGrid (blue) are almost overlapped in row 1, 2) the lower the NLL, the better the model is.

As shown in Figure 7, although BHMs are also not scalable as VSDGPOMs, SBHMs are perfectly scalable. In general, accuracy of both BHMs and SBHMs are slightly better than VSDGPOMs while significantly better than DGrid. Although DGPOM and DGPOM50% seem to have a better NLL at first, they cannot be run for more than approximately 50 time steps because of the unwieldy growing computational time. The average time to update the SBHM model for each new scan are 450, 8, and 970 ms for datasets 1,2, and 3, respectively. Most importantly, SBHMs have a approximately constant update time similar to that of occupancy Grid maps, yet bringing all advantages of other continuous mapping techniques.

7 Conclusions

We extended the Hilbert maps algorithm for mapping long-term dynamics. The variational Bayesian formulation is fast for use in real time and highly scalable. We eliminated some vital parameter tuning in conventional Hilbert maps, making it further convenient to use. Additionally, we demonstrated that the maps are less susceptible to occlusions as they consider neighborhood information. These inherent properties in the proposed approach as well as the main components of the algorithm—kernels and Bayesian learning—can form the basis for developing a real-time simultaneous mapping and path planning algorithm under a single framework.

References

[1] S. Scheding, G. Dissanayake, E. M. Nebot, and H. Durrant-Whyte. An experiment in autonomous navigation of an underground mining vehicle. *IEEE Transactions on Robotics and Automation*, 15(1):85–95, 1999.

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

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

[4] J. Wang and B. Englot. Fast, accurate gaussian process occupancy maps via test-data octrees and nested bayesian fusion. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1003–1010, 2016.

[5] M. Norouzii, J. V. Miro, and G. Dissanayake. Probabilistic stable motion planning with stability uncertainty for articulated vehicles on challenging terrains. *Autonomous Robots*, 40(2): 361–381, 2016.

[6] M. Turchetta, F. Berkenkamp, and A. Krause. Safe exploration in finite markov decision processes with gaussian processes. In *Advances in Neural Information Processing Systems (NIPS)*, pages 4312–4320, 2016.

[7] G. Francis, L. Ott, and F. Ramos. Stochastic functional gradient for motion planning in continuous occupancy maps. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[8] Z. Marinho, B. Boots, A. Dragan, A. Byravan, G. J. Gordon, and S. Srinivasa. Functional gradient motion planning in reproducing kernel hilbert spaces. In *Proceedings of Robotics: Science and Systems (RSS)*, 2016.

[9] R. Senanayake, S. O'Callaghan, and F. Ramos. Learning highly dynamic environments with stochastic variational inference. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2017.

[10] F. Ramos and L. Ott. Hilbert maps: scalable continuous occupancy mapping with stochastic gradient descent. In *Proceedings of Robotics: Science and Systems (RSS)*, Rome, Italy, July 2015.

[11] K. Doherty, J. Wang, and B. Englot. Probabilistic map fusion for fast, incremental occupancy mapping with 3d hilbert maps. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 0–0, 2016.

[12] V. Guizilini and F. Ramos. Learning to reconstruct 3d structures for occupancy mapping. In *Proceedings of Robotics: Science and Systems (RSS)*, 2017.

[13] R. Senanayake, L. Ott, S. O'Callaghan, and F. Ramos. Spatio-temporal hilbert maps for continuous occupancy representation in dynamic environments. In *Neural Information Processing Systems (NIPS)*, 2016.

[14] J. Saarinen, H. Andreasson, and A. Lilienthal. Independent Markov Chain Occupancy Grid Maps for Representation of Dynamic Environment. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3489–3495, 2012.

[15] D. Meyer-Delius, M. Beinhofer, and W. Burgard. Occupancy Grid Models for Robot Mapping in Changing Environments. In *proc. AAAI Conference on Artificial Intelligence (AAAI)*, pages 2024–2030, 2012.

[16] Z. Wang, P. Jensfelt, and J. Folkesson. Modeling spatial-temporal dynamics of human movements for predicting future trajectories. In *AAAI Conference on Artificial Intelligence*, pages 42–48, 2015.

[17] N. C. Mitsou and C. Tzafestas. Temporal Occupancy Grid for mobile robot dynamic environment mapping. In *Mediterranean Conference on Control & Automation (MED)*, pages 1–8, 2007.

[18] T. Krajnık, J. P. Fentanes, G. Cielniak, C. Dondrup, and T. Duckett. Spectral analysis for long-term robotic mapping. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3706–3711, 2014.

[19] D. Hahnel, R. Triebel, W. Burgard, and S. Thrun. Map building with mobile robots in dynamic environments. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 4270–4275, 2003.

[20] A. Walcott. *Long-term robot mapping in dynamic environments*. PhD thesis, Massachusetts Institute of Technology, 2011.

[21] R. Senanayake and F. Ramos. Bayesian hilbert maps for continuous occupancy mapping in dynamic environments. In *Workshop on Machine Learning for Autonomous Vehicles at the 34th International Conference on Machine Learning (ICML)*, 2017.

[22] C. M. Bishop. Pattern recognition. *Machine Learning*, 128:1–58, 2006.

[23] T. Jaakkola and M. Jordan. A variational approach to bayesian logistic regression models and their extensions. In *Sixth International Workshop on Artificial Intelligence and Statistics*, volume 82, 1997.

[24] R. Ranganath, S. Gerrish, and D. Blei. Black box variational inference. In *Artificial Intelligence and Statistics (AISTATS)*, pages 814–822, 2014.

[25] D. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. In *arXiv:1601.00670v5*, 2017.

[26] C. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

[27] S. T. O'Callaghan and F. T. Ramos. Gaussian process occupancy maps for dynamic environments. In *The 14th International Symposium on Experimental Robotics (ISER)*, Marrakesh, Morocco, June 2014.

[28] D. Arbuckle, A. Howard, and M. Mataric. Temporal occupancy grids: a method for classifying the spatio-temporal properties of the environment. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, volume 1, pages 409–414, 2002.

Bayesian Hilbert Maps for Dynamic Continuous Occupancy Mapping - supplementary materials

Ransalu Senanayake
School of Information Technologies
The University of Sydney
Australia
rsen4557@uni.sydney.edu.au

Fabio Ramos
School of Information Technologies
The University of Sydney
Australia
fabio.ramos@sydney.edu.au

1 Demonstrating the concept behind sequential Bayesian Hilbert maps (SBHMs) using Bayesian linear regression

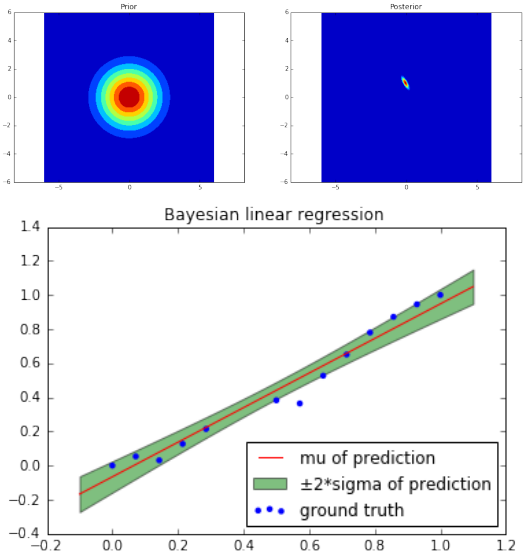


Figure 1: a

In Bayesian linear regression, a diffused prior indicated by the isotopic Gaussian can be used to build a shrunk posterior. That posterior can be used for making predictions (mean and variance).
Let us consider a uniform prior: $\mathbf{m}_0 = \mathbf{0}$ and $\mathbf{S}_0 = \alpha^{-1}\mathbf{I}$

$$p(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_0, \mathbf{S}_0) \tag{1}$$

Then, the posterior is,

$$p(\mathbf{w}|\mathbf{t}) = \mathcal{N}(\mathbf{w}|\mathbf{m}_N, \mathbf{S}_N) \tag{2}$$

where, $\mathbf{m_N} = \beta \mathbf{S}_N \mathbf{\Phi}_\mathbf{x}^\top \mathbf{t}$, and $\mathbf{S}_N = (\alpha\mathbf{I} + \beta {\mathbf{\Phi}_\mathbf{x}}^\top \mathbf{\Phi}_\mathbf{x})^{-1}$

Learning can be done sequentially, considering one or a few data points at a time and updating the posterior. Once the posterior is updated, data can in fact be discarded as the posterior retains information. Therefore, the model can be updated as new data are arrived sequentially without accumulating data. The procedure is illustrated in Figure 2. In contrast to this method, SBHMs,

1. use a logistic regression classifier instead of linear regression

2. work in rich feature space instead of using raw data

1st Conference on Robot Learning (CoRL 2017), Mountain View, United States.

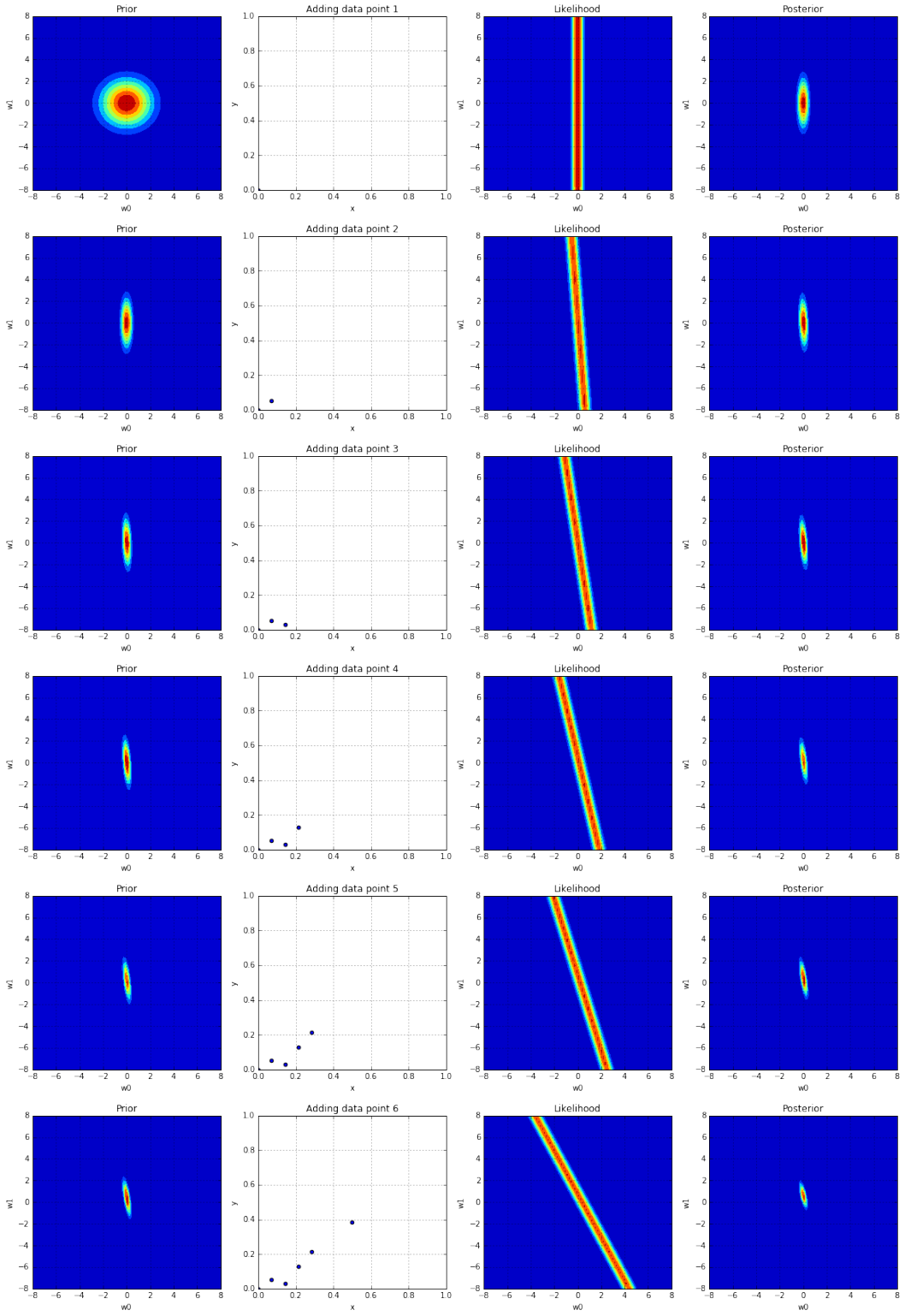


Figure 2: Consider the first row. Column1: We start with an isotropic prior with a large variance. Column 2: A data point $(0, 0)$ is added. Column 3: The likelihood for that data point is computed. Column 4: By multiplying the prior with the likelihood and then normalizing, the posterior is obtained. Now consider the second row. Same procedure is followed, having the posterior from the previous step used as the new prior and multiplying with the likelihood of the new data point. This way, new information from each step is absorbed into the posterior by restricting the posterior sequentially.

2 Learning parameters

The equations used in Algorithm 1 and 2 are detailed here.

2.1 BHMs

The objective function of the model is,

$$\tilde{\mathcal{L}}(\mathrm{Q},\boldsymbol{\xi}) = \frac{1}{2}\log|\boldsymbol{\Sigma}| + \frac{1}{2}\mu\boldsymbol{\Sigma^{-1}}\mu^{\top} + \sum_{n=0}^{t}\left(\log\sigma(\xi_n) + \xi_n^2\lambda(\xi_n) - \frac{\xi_n}{2}\right) + \log\frac{\Gamma(a)}{\Gamma(a_0)} + \log\frac{{b_0}^{a_0}}{b^a} + a\left(1-\frac{b_0}{b}\right) \quad (3)$$

, where,

$$\lambda(\xi) = \frac{1}{2\xi}\Big(\sigma(\xi) - \frac{1}{2}\Big), \quad (4)$$

with ξ as a local parameter used to linearize the function using the Taylor expansion.

E-step: Estimating $\mathrm{Q}(\mathbf{w}) = \mathcal{N}(\mathbf{w}|\mu,\boldsymbol{\Sigma})$,

$$\boldsymbol{\mu} = \Sigma\left(\Sigma_0^{-1}\boldsymbol{\mu}_0 + \sum_{n=1}^{t}(y_n - 0.5)\Phi(\mathbf{x}_n)\right) \quad (5)$$

$$\boldsymbol{\Sigma}^{-1} = \left\{\frac{a}{b_d}\mathbf{I}_{dd}\right\}_{d=1}^{D} + 2\sum_{n=1}^{N} = t\lambda(\xi_n)\Phi(\mathbf{x}_n)\Phi(\mathbf{x}_n)^{\top} \quad (6)$$

Estimating $\mathrm{Q}(\boldsymbol{\alpha}) = \Gamma(\boldsymbol{\alpha}|a,\mathbf{b}) = \prod_{d=1}^{D}\Gamma(\alpha_d|a,b_d)$,

$$a = a_0 + \frac{1}{2} \quad (7)$$

$$b_d = b_0 + \frac{1}{2}\mathbf{w}_d^2 + \Sigma_{dd} \quad (8)$$

M-step:

$$\xi_k^2 = \Phi^{\top}(\mathbf{x}_k)(\Sigma_t + \boldsymbol{\mu}_t\boldsymbol{\mu}_t^{\top})\Phi(\mathbf{x}_k) \quad (9)$$

Here, t is the current time step. All data points from time$=0$ to time$=t$ are in each iteration.

2.2 SBHMs

The objective function is,

$$\mathcal{L}(\mathrm{Q},\boldsymbol{\xi}) = \frac{1}{2}\log\left|\frac{\Sigma_t}{\Sigma_0}\right| + \frac{1}{2}\boldsymbol{\mu}_t^{\top}\Sigma_t^{-1}\boldsymbol{\mu}_t - \frac{1}{2}\boldsymbol{\mu}_0^{\top}\Sigma_0^{-1}\boldsymbol{\mu}_0 + \sum_{k=0}^{K_t}\left(\log\sigma(\xi_k) + \xi_k^2\lambda(\xi_k) - \frac{\xi_k}{2}\right) \quad (10)$$

E-step:

$$\boldsymbol{\mu}_t = \Sigma_t\left(\Sigma_{t-1}^{-1}\boldsymbol{\mu}_{t-1} + \sum_{k=1}^{K_t}(y_k - 0.5)\Phi(\mathbf{x}_k)\right) \quad (11)$$

$$\Sigma_t^{-1} = \Sigma_{t-1}^{-1} + 2\sum_{k=1}^{K_t}\lambda(\xi_k)\Phi(\mathbf{x}_k)\Phi^{\top}(\mathbf{x}_k) \quad (12)$$

Here, k indicates all the points in the t^{th} time scan. There are K_t such data points.

M-step:

$$\xi_k^2 = \Phi^{\top}(\mathbf{x}_k)(\Sigma_t + \boldsymbol{\mu}_t\boldsymbol{\mu}_t^{\top})\Phi(\mathbf{x}_k) \quad (13)$$

For a more detailed discussion on the topic, readers are encouraged to go through [22] and [25].

2.3 Practical considerations for the implementation

For numerical stability perform inversions $A\mathbf{x} = \mathbf{b} \Rightarrow \mathbf{x} = A^{-1}\mathbf{b}$ using Cholesky factorization $A = LL^*$.

3 Experiments

3.1 Initial settings

In order to have an approximately diffuse priors, the initial conditions of the BHM were $a_0 = 10^{-3}$ and $b_0 = 10^{-4}$ while they were $\sigma_0 = 10^{-4}$ in SBHM. The filter threshold was always $\eta = 0.3$. The hinging locations $\tilde{\mathbf{x}}$ were set every 5 m for datasets 1 and 2 while they were set every 2 m for dataset 3. Theoretically, the higher the number of centers, the richer the feature space, and slower the algorithm will be.

3.2 Other observations

In SBHMs, although the very first step would take around 10 iterations for convergence, merely one iteration is sufficient for subsequent model updates.