Stochastic Meta Descent in Online Kernel Methods

Supawan Phonphitakchai
Department of Electrical and Computer Engineering
Naresuan University, Thailand
supawanph@nu.ac.th

Tony J. Dodd
Department of Automatic Control and Systems Engineering
The University of Sheffield, UK
t.j.dodd@sheffield.ac.uk

Abstract—**Learning System is a method to approximate an underlying function from a finite observation data. Since batch learning has a disadvantage in dealing with large data set, online learning is proposed to prevent the computational expensive. Iterative method called Stochastic Gradient Descent (SGD) is applied to solve for the underlying function on reproducing kernel Hilbert spaces (RKHSs). To use SGD in time-verying environment, a learning rate is adjusted by Stochastic Meta Descent (SMD). The simulation results show that SMD can follow shifting and switching target function whereas the size of model can be restricted using sparse solution.**

I. Introduction

The objective of learning is to approximate an underlying function from an observation data perturbed with noise. Once the unknown function is accurately estimated, it can be applied to predict future from a given input. Generally, batch learning is assumed but this is very inefficient in the case of large data set. To overcome this problem, online learning can be employed. This framework uses only one sample at a time and then discards after learning. Therefore, online learning do not need to store the whole set of sample.

Reproducing kernel Hilbert spaces (RKHSs) show advantages in solving a number of well-known problems in signal processing, control, machine learning and function approximation. The strong theoretical foundations of RKHS are developed to find the global and usually unique solutions, since it uses the idea of convex optimisation in training method [1]. Recent interest in kernel methods has been dominated in machine learning community i.e. support vector machines, Gaussian processes and regularisation networks.

The first contribution of this work is to present an online learning in RKHS by investigating the approximation function from a given data (supervise learning) which is generated from time-varying function. The iterative method, called Stochastic Gradient Descent (SGD), is applied to solve the approximation functions in non-stationary environment by adjusting the learning rate using the idea of Stochastic Meta Descent (SMD). The other contribution is to restrict the size of approximation function by sparse solution using orthogonal projection method in non-stationary environment.

The next section provides the general framework of RKHS followed by SGD method to investigate the approximation function. SMD to adjust the learning rate and the method to bound the size of model are shown in Sec IV and V respectively. Finally, the simulations in Sec VI show results of using SMD to model a time-varying function which are illustrated in two examples and the paper ends with the conclusions.

II. Preliminaries and Notation

A set of samples is given in a form of observation data $\{x_i, z_i\}_{i=1}^N \in X \times Z$. The unknown $f : X \times Z \rightarrow R$ is assumed to belong to some RKHS, $\mathcal{H}_k$, defined on X whereas the space of all possible observations is Z. Neglecting errors, the observations arise as follows

$$z_i = L_i f \tag{1}$$

where $\{L_i\}_{i=1}^N$ is a set of linear evaluation functionals which has a unique correspondence to z_i. The complete set of z_i can be expressed by [2]

$$\mathbf{z} = \sum_{i=1}^{N} (L_i f) e_i = Lf \tag{2}$$

where $e_i \in R^N$ is the ith standard basis vector.

A RKHS can be defined as a Hilbert space of functions on X, with the property that, for each $x \in X$, the evaluation functional, L_i, which associates f with $f(x_i)$, $L_i f \rightarrow f(x_i)$, is a bounded linear functional. The boundedness means that there exists a scalar M such that

$$|L_i f| = |f(x_i)| \leq M \|f\|_{\mathcal{H}_k} \quad \text{for all } f \text{ in the RKHS} \tag{3}$$

where $\| \cdot \|_{\mathcal{H}_k}$ is the norm in RKHS [3]. Following from the Riesz representation theorem, the observations are given by [3]

$$L_i f = \langle k_i, f \rangle = f(x_i) \qquad f \in \mathcal{H}_k \tag{4}$$

where k_i depends only on x_i. Therefore the learning problem can be stated as follows: given RKHS of functions ($\mathcal{H}_k$), the set of functions $\{k_i\}_{i=1}^N \in \mathcal{H}_k$, and the observations $\{x_i, z_i\}_{i=1}^N$, find a function $f \in \mathcal{H}_k$, such that (4) is satisfied.

The k_i is a positive definite function called reproducing kernel (r.k.) of $\mathcal{H}_k$ which is the unique representation of evaluation at x_i [4]. The function $k : X \times X$ can be defined by Riesz theorem such that

$$\langle k(x_i, \cdot), k(x_j, \cdot) \rangle_{\mathcal{H}_k} = k(x_i, x_j) \tag{5}$$

where $k(x_i, \cdot)$ is a function at X centered on x_i. There are several examples of kernel function i.e. polynomial, Gaussian and sigmoid kernel. We can then represent any functions with reproducing kernel k in the form

$$f(\cdot) = \sum_{i=1}^{N} \alpha_i k(x_i, \cdot) \tag{6}$$

for $N \in \mathbb{Z}^+$ and $\alpha_i \in \mathbb{R}$. This expression defines a finite dimensional subspace of $\mathcal{H}_k$ and r.k. are a basis for RKHS. The learning problem then reduces to that of estimating the parameters, α_i, in (6).

III. Function Approximation with SGD

In online learning, at each iteration, we observe only a part of Z, denoted z_n (typically the nth observation). The associated linear evaluation functional at present time is then [5]

$$L_n f = z_n. \tag{7}$$

The function approximation of f at time n can be achieved by minimising the instantaneous non-negative functional

$$\hat{g}_{reg}(f_n) = \frac{1}{2}\|L_{n+1}f_n - z_{n+1}\|^2 + \frac{\rho}{2}\|f_n\|^2 \tag{8}$$

where $\frac{\rho}{2}\|f_n\|^2$ is a regularisation term and $\rho \geq 0$ is a regularisation parameter. The term $\frac{\rho}{2}\|f_n\|^2$ is added to prevent ill-posed condition that the estimation function fits very well on observation data but cannot fit the unforeseen data.

The minimisation of (8) can be determined by using SGD such that the function update, f_{n+1}, will be calculated at each incoming observations, (x_n, z_n), by

$$f_{n+1} = f_n - \eta_n \nabla \hat{g}_{reg}(f_n) \tag{9}$$

where $\nabla \hat{g}_{reg}(f_n)$ is the instantaneous gradient of $\hat{g}_{reg}$ or the direction of gradient descent at f_n and η_n is the learning rate. Hence, we obtain the function update

$$f_{n+1} = (1 - \eta_n \rho_n) f_n - \eta_n L^*_{n+1}(L_{n+1}f_n - z_{n+1}) \tag{10}$$

where L^* is the adjoint operator of L defined by $\langle Lf, z\rangle = \langle f, L^* z\rangle$. For some constant, $L^*_{n+1}a = k_{n+1}a$ and also $L^*_{n+1}f_n = f_n(x_{n+1})$, therefore

$$f_{n+1} = (1 - \eta_n \rho_n) f_n - \eta_n k_{n+1}(f_n(x_{n+1}) - z_{n+1}). \tag{11}$$

From (6), the function update can be written in the form of kernel function given by

$$\begin{aligned} f_{n+1}(x) &= (1 - \eta_n \rho_n) \sum_{i=1}^{n} \alpha_n^i k_i(x) - \eta_n e_{n+1} k_{n+1}(x) \\ &= \sum_{i=1}^{n+1} \alpha_{n+1}^i k_i(x) \end{aligned} \tag{12}$$

where the prediction error $e_{n+1} = L_{n+1}f_n - z_{n+1}$ and $L^*_{n+1}e_{n+1} = k_{n+1}e_{n+1}$. Hence, the parameters α^i_{n+1} are updated from

$$\alpha^i_{n+1} = \begin{cases} (1 - \eta_n \rho)\alpha^i_n & \text{for } i \leq n \\ -\eta_n e_{n+1} & \text{for } i = n+1. \end{cases} \tag{13}$$

In SGD, the learning rate, η_n, plays an important role as the step size moving in the direction that $\hat{g}_{reg}(f_n)$ is minimised. The value of the learning rate to be used is critical. Small values can lead to a slow minimisation whereas high values may cause divergence. Moreover, using SGD in time-varying systems, the learning rate has to be adapted online. Based on this idea, we propose SMD to adjust learning rate which can model the unknown system in time-varying environments.

IV. Learning Rate Adaptation using SMD

This methods avoids negative values of the learning rate by making updates via an exponential gradient algorithm [6], [7]. Also, all past gradients are taken into account for updating the present parameter which means that information of every past data is incorporated in the update rule.

The local learning rate is best adapted by exponential gradient descent, so that it can cover the wide dynamic range whilst staying strictly positive [6]. The learning rate is updated as the followings

$$\begin{aligned} \ln \eta_n &= \ln \eta_{n-1} - \mu \frac{\partial \hat{g}_{reg}(f_n)}{\partial \ln \eta_{n-1}} \\ &= \ln \eta_{n-1} - \mu \frac{\partial \hat{g}_{reg}(f_n)}{\partial f_n} \cdot \frac{\partial f_n}{\partial \ln \eta_{n-1}} \end{aligned} \tag{14}$$

and therefore

$$\eta_n = \eta_{n-1} \cdot \exp(-\mu \langle h_n, v_n \rangle) \tag{15}$$

where $h_n \equiv \frac{\partial \hat{g}_{reg}(f_n)}{\partial f_n}$, $v_n \equiv \frac{\partial f_n}{\partial \ln \eta_{n-1}}$ and μ is a global meta learning rate.

To make the computation simpler, the linearization $e^u \approx 1+u$ is used, valid for small $|u|$ [7]. Hence, the update rule for the learning rate is

$$\eta_n = \eta_{n-1} \cdot \max(\frac{1}{2}, 1 - \mu\langle h_n, v_n \rangle). \tag{16}$$

The gradient trace, v, measures the effect that the learning rate makes on the function update. Here, we consider the effect of all changes up to time n in the form of the exponential average. Therefore, the gradient trace is given by the summation of gradient up to time n [7]

$$v_{n+1} \equiv \sum_{i=0}^{n} \lambda^i \frac{\partial f_{n+1}}{\partial \ln \eta_{n-i}} \tag{17}$$

where $\lambda \in [0, 1]$ is a forgetting factor. Substituting f_{n+1} given in (9)

$$\begin{aligned} v_{n+1} &= \sum_{i=0}^{n} \lambda^i \frac{\partial f_n}{\partial \ln \eta_{n-i}} - \sum_{i=0}^{n} \lambda^i \frac{\partial(\eta_n \cdot h_n)}{\partial \ln \eta_{n-i}} \\ &\approx \lambda v_n - \eta_n \cdot h_n - \eta_n \sum_{i=0}^{n} \lambda^i \frac{\partial h_n}{\partial f_n} \cdot \frac{\partial f_n}{\partial \ln \eta_{n-i}} \\ &= \lambda v_n - \eta_n (h_n + \lambda H_n v_n) \end{aligned} \tag{18}$$

where H_n denotes the Hessian of $\hat{g}_{reg}(f_n)$ at time n [7].

Considering the update rule in (16), we need to define the dot product of h_n and v_n. Here,

$$\begin{aligned} h_n &= \frac{\partial \hat{g}_{reg}(f_n)}{\partial f_n} = \nabla \hat{g}_{reg}(f_n) \\ &= L^*_{n+1}(L_{n+1}f_n - z_{n+1}) + \rho_n f_n. \end{aligned} \tag{19}$$

Hessian matrix H is given by

$$H_n = \frac{\partial h_n}{\partial f_n} = \frac{\partial^2 \hat{g}_{reg}(f_n)}{\partial f_n^2} = L^*_{n+1}L_{n+1} + \rho. \tag{20}$$

Therefore, the update rule for the gradient trace given in (18) is equivalent to

$$\begin{aligned} v_{n+1} & \\ &= \lambda v_n - \eta_n(L^*_{n+1}e_{n+1} + \rho f_n) - \eta_n\lambda(L^*_{n+1}L_{n+1} + \rho)v_n \\ &= (1-\rho\eta_n)\lambda v_n - \eta_n\rho f_n - \eta_n\lambda k_{n+1}v_n(x_{n+1}) \\ &\quad -\eta_n k_{n+1}e_{n+1}. \end{aligned} \tag{21}$$

The gradient trace can also be represented in the form of r.k. as

$$v_{n+1} = \sum_{i=1}^{n+1} \omega^i_{n+1} k_i \tag{22}$$

where $\omega_i \in \mathbb{R}$. Combining (21) and (22), the gradient trace in the form of reproducing kernels and its parameters are presented by

$$\begin{aligned} v_{n+1} &= (1-\rho\eta_n)\lambda \sum_{i=1}^{n} \omega^i_n k_i - \eta_n\rho \sum_{i=1}^{n} \alpha^i_n k_i \\ &\quad -\eta_n k_{n+1}(\lambda v_n(x_{n+1}) - e_{n+1}) \end{aligned} \tag{23}$$

and the parameter vector ω_{n+1} has an update

$$\omega^i_{n+1} = \begin{cases} (1-\eta_n\rho)\lambda\omega^i_n - \eta_n\rho\alpha^i_n & \text{for } i \le n \\ -\eta_n\lambda v_n(x_{n+1}) - \eta_n e_{n+1} & \text{for } i = n+1. \end{cases} \tag{24}$$

Therefore, the inner product $\langle h_n, v_n\rangle$ is as the followings

$$\begin{aligned} \langle h_n, v_n\rangle &= \langle L^*_{n+1}e_{n+1} + \rho f_n, v_n\rangle \\ &= e_{n+1}v_n(x_{n+1}) + \rho\alpha^T_n K\omega_n \end{aligned} \tag{25}$$

where $\langle f_n, v_n\rangle = \langle \sum_{i=1}^n \alpha^i_n k_i, \sum_{i=1}^n \omega^i_n k_i\rangle = \alpha^T_n K\omega_n$. Hence, the update rule for the learning rate is

$$\eta_n = \eta_{n-1} \cdot \max(\frac{1}{2}, 1 - \mu(e_{n+1}v_n(x_{n+1}) + \rho\alpha^T_n K\omega_n)). \tag{26}$$

V. Sparse Solution

Considering the framework of function approximation presented in Section III, in each iteration, the posterior function update is added with a new kernel and parameters. It obviously expresses that the size of function grows without bound. To overcome this problem, we need to restrict the model growth by [5]

1) adding only kernels that significantly reduce the approximation error (incremental step) and
2) removing kernels that no longer create significant error (decremental step).

The method called orthogonal projections will be applied to do these.

A. Incremental Step

The function update, f_n, belongs to RKHS and is made up of n kernels with linear combination which lies in n dimensional subspace, $f_n \in \mathcal{H}_{k_n}$. In the same way, adding a new kernel to f_n, the new function update belongs to subspace $\mathcal{H}_{k_{n+1}}$ such that $f_{n+1} \in \mathcal{H}_{k_{n+1}}$. However, the updating can be done without adding a new kernel by using orthogonal projection that function f_{n+1} is projected onto $\mathcal{H}_{k_n}$ denoted by $f^\perp_{n+1}$. The orthogonal projection, $f^\perp_{n+1}$, takes an effect of the new data point but the size of model is maintained such that the number of kernel term is n.

The orthogonal projection $f^\perp_{n+1}$ will be chosen when a square norm of difference, $\|f_{n+1} - f^\perp_{n+1}\|$ is lower than the priori defined threshold, κ_{inc}, otherwise the update function becomes f_{n+1}.

Considering the orthogonal projection of f_{n+1} onto $\mathcal{H}_{k_n}$, we then have $f_{n+1} - f^\perp_{n+1} \perp \mathcal{H}_{k_n}$ [8]. Defining $f_{n+1} = \sum_{l=1}^{n+1}\alpha^l_{n+1}k_l$ which vector parameter α_{n+1} has been previously determined in approximating f_{n+1}, and $f^\perp_{n+1} = \sum_{j=1}^{n}\beta^j_{n+1}k_j$. Using projection conditions that $\langle k_i, f_{n+1} - f^\perp_{n+1}\rangle = 0$ give

$$\langle k_i, \sum_{l=1}^{n+1}\alpha^l_{n+1}k_l - \sum_{j=1}^{n}\beta^j_{n+1}k_j\rangle = 0 \tag{27}$$

where $i = 1, \dots, n$ or equivalent to

$$\begin{aligned} &\alpha^1_{n+1}\langle k_i, k_1\rangle + \dots + \alpha^{n+1}_{n+1}\langle k_i, k_{n+1}\rangle - \\ &\quad \beta^1_{n+1}\langle k_i, k_1\rangle - \dots - \beta^n_{n+1}\langle k_i, k_n\rangle = 0 \end{aligned} \tag{28}$$

where $i = 1, \dots, n$. The above expression has the unique solution as the following

$$\beta_{n+1} = K^{-1}_{n,n}c. \tag{29}$$

Note that $K_{n,n} \in \mathbb{R}^{n,n}$ and $c_i = \alpha^1_{n+1}\langle k_i, k_1\rangle + \dots + \alpha^{n+1}_{n+1}\langle k_i, k_{n+1}\rangle$ where $i = 1\dots, n$.

Considering the gradient trace v_{n+1} in (22), the orthogonal projection of the gradient trace has to be used if $f^\perp_{n+1}$ is selected. This can be achieved using the same methods as given in (27)-(29). Defining $v^\perp_{n+1} = \sum_{j=1}^n \psi^j_{n+1}k_j$, the parameter ψ_{n+1} is equivalent to

$$\psi_{n+1} = K^{-1}_{n,n}c' \tag{30}$$

where $c'_i = \omega^1_{n+1}\langle k_i, k_1\rangle + \dots + \omega^{n+1}_{n+1}\langle k_i, k_{n+1}\rangle$ and $i = 1\dots, n$. The square norm of difference between f_{n+1} and $f^\perp_{n+1}$ which is given by

$$\begin{aligned} \|f_{n+1} - f^\perp_{n+1}\|^2 &= \langle f_{n+1} - f^\perp_{n+1}, f_{n+1} - f^\perp_{n+1}\rangle \\ &= \langle f_{n+1}, f_{n+1} - f^\perp_{n+1}\rangle - 0. \end{aligned} \tag{31}$$

The second term of RHS of (31) disappears due to the condition of orthogonal projection. Therefore

$$\begin{aligned} &\|f_{n+1} - f^\perp_{n+1}\|^2 \\ &= \sum_{l=1}^{n+1}\sum_{j=1}^{n+1}\alpha^l_{n+1}\alpha^j_{n+1}\langle k_l, k_j\rangle - \sum_{l=1}^{n+1}\sum_{j=1}^{n}\alpha^l_{n+1}\beta^j_{n+1}\langle k_l, k_j\rangle \\ &= \alpha^T_{n+1}K_{n+1,n+1}\alpha_{n+1} - \alpha^T_{n+1}K_{n+1,n}\beta_{n+1} \end{aligned} \tag{32}$$

where $\alpha_{n+1}, \beta_{n+1}$ are the vector of parameters, $K_{n+1,n+1} \in \mathbb{R}^{n+1,n+1}$ and $K_{n+1,n} \in \mathbb{R}^{n+1,n}$.

B. Decremental Step

After the incremental step, we denote the new function update by

$$f^\uparrow_{n+1} = \sum_{j=1}^{m}\nu^j_{n+1}k_j \tag{33}$$

where m can be n or $n+1$ and ν_{n+1} is a vector equal to α_{n+1} or β_{n+1} as appropriate when the function update is f_{n+1} or $f^\perp_{n+1}$ consequently. The size of existing model is reduced by eliminating an unwanted kernel term. Defining the function called decremental estimation with lth kernel removed by

$$f_{n+1}^{\downarrow\backslash l} = \sum_{j\in\mathcal{M}} \zeta_{n+1}^{j} k_j \tag{34}$$

where $l \in m$ and $\mathcal{M} = \{1, 2, \ldots, m\}\backslash l$. The function $f_{n+1}^{\downarrow\backslash l}$ is an orthogonal projection of $f_{n+1}^{\uparrow}$ onto the reduced set of kernels forming a subspace $\mathcal{H}_{k_{\mathcal{M}}}$. The kernel term l will be removed when the square norm difference, $\|f_{n+1}^{\uparrow} - f_{n+1}^{\downarrow\backslash l}\|^2$, is lower than the threshold κ_{dec} and the current model then becomes $f_{n+1}^{\downarrow\backslash l}$. Using the same methods given in the previous section, the ζ_{n+1} can be estimated by using the projection conditions. Therefore,

$$\zeta_{n+1} = K_{\mathcal{M},\mathcal{M}}^{-1} d. \tag{35}$$

Note that, $K_{\mathcal{M},\mathcal{M}}^{-1} \in \mathbb{R}^{m-1,m-1}$ and $d_j = \nu_{n+1}^{1}\langle k_1, k_j\rangle + \ldots + \nu_{n+1}^{m}\langle k_m, k_j\rangle$ where $j \in \mathcal{M}$. The norm difference is given as follows

$$\|f_{n+1}^{\uparrow} - f_{n+1}^{\downarrow\backslash l}\|^2 = \nu_{n+1}^{T} k_{m,m} \nu_{n+1} - \nu_{n+1}^{T} K_{m,\mathcal{M}} \zeta_{n+1} \tag{36}$$

where $k_{m,m} \in \mathbb{R}^{m,m}$ and $K_{m,\mathcal{M}} \in \mathbb{R}^{m,m-1}$.

If the kernel lth is removed for the function update, the gradient trace needs to be removed at the term l as well. Defining $v_{n+1}^{\uparrow} = \sum_{j=1}^{m} \chi_{n+1}^{j} k_j$ for the gradient trace such that χ_{n+1} can be ω_{n+1} or ψ_{n+1} as appropriate. The decremental estimation when kernel lth removed is $v_{n+1}^{\downarrow\backslash l} = \sum_{j\in\mathcal{M}} \varphi_{n+1}^{j} k_j$ and the parameters φ_{n+1} is determined by

$$\varphi_{n+1} = K_{\mathcal{M},\mathcal{M}}^{-1} d' \tag{37}$$

where $d' = \chi_{n+1}^{1}\langle k_1, k_j\rangle + \ldots + \chi_{n+1}^{m}\langle k_m, k_j\rangle$ and $j \in \mathcal{M}$.

VI. Simulated Evaluation of the Algorithms

SMD were evaluated with observation data from non-stationary environments which was generated from a sinc function $f(x) = \frac{sin\pi(x-x')}{\pi(x-x')}$ where x' is the centre. The experimental data was corrupted by Gaussian distributed noise with $N(0, \sigma^2)$. Throughout our experiments, σ^2 was setting to 0.1. To simulate non-stationary, our sinc function was varied in time in two cases; the center was shifted with rate $0.0004n$ and switched to other value at the 150th and 300th iterations.

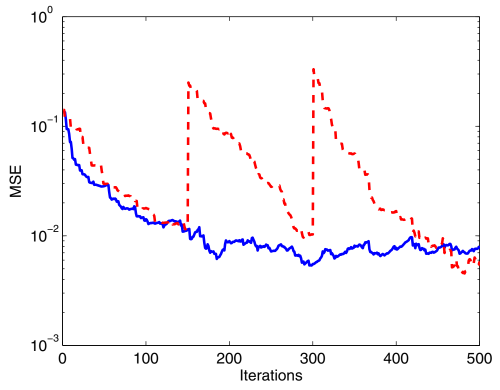
In our experiments, we used 500 samples to test SMD algorithm with Gaussian kernel and width equal to 50. A test set of 100 data points were used to verify the performance of the algorithms in terms of MSE, where

$$MSE = \frac{1}{N}\sum_{n=1}^{N}(f(x_n) - z_n)^2. \tag{38}$$

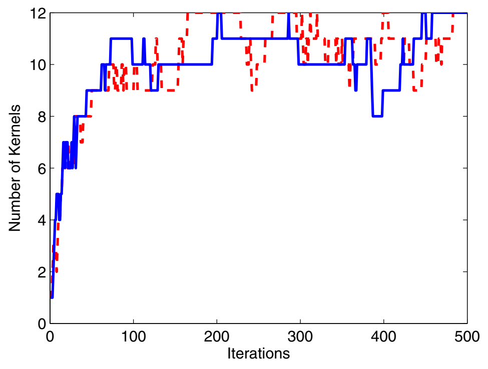
Fig 1(a) presents MSE of applying SMD in shifting and switching target, the results show that our algorithm can follow the target even they are time-varying function. The experiments also include sparse solution to bound the size of model. They are illustrated in Fig 1(b) that the number of kernels was reduced from 500 to around 12 and the time of program running is faster.

VII. Conclusions

This work presents a framework for solving the approximation function on RKHS by using online SGD. For applying in time-varying environment, the learning rate is adapted at each iteration by SMD whereas the regularisation parameter is kept constant. The simulation results support that using SMD is able to track the time-varying function in two cases: shifting and switching target function. The size of model is bound by sparse solution which can prevent the computational expensive and decrease the time of program running.



(a)



(b)

Fig. 1. Variation in (a) MSE and (b) number of kernels in SMD with sparse solution by setting $\kappa_{inc} = \kappa_{dec} = 0.005$, $\mu = 0.1$, $\lambda = 0.99$ and $\eta_0 = 0.1$. Shown are shifting ('—') and switching target function ('– –').

References

[1] C. Campbell, *Radial Basis Function Networks: Design and Applications*. Springer Verlag, 2000.
[2] T. Dodd and R. Harrison, "Some lemmas on reproducing kernel Hilbert spaces," Department of Automatic Control and Systems Engineering, University of Sheffield, UK, Tech. Rep. 819, 2002.
[3] N. Akhiezer and I. Glazman, *Theory of Linear Operators in Hilbert Space*. Pitman, 1981, vol. I.
[4] G. Wahba, *Spline Models for Observational Data*, ser. Series in Applied Mathematics. Philadelphia: SIAM, 1990, vol. 50.
[5] T. Dodd, B. Mitchinson, and R. Harrison, "Sparse stochastic gradient descent learning in kernel models," in *Proceedings of The Second International Conference on Computational Intelligence, Robotics and Autonomous Systems, Singapore*, 2003.
[6] N. N. Schraudolph and T. Graepel, "Towards stochastic conjugate gradient methods," in *Proc. 9th Intl. Conf. Neural Information Processing*, L. Wang, J. C. Rajapakse, K. Fukushima, S.-Y. Lee, and X. Yao, Eds. IEEE, 2002, pp. 853–856.
[7] S. V. N. Vishwanathan, N. N. Schraudolph, and A. J. Smola, "Step size adaptation in reproducing kernel Hilbert space," *J. Mach. Learn. Res.*, vol. 7, pp. 1107–1133, 2006.
[8] E. Kreyszig, *Introductory Functional Analysis with Applications*. John Wiley & Sons, 1978.