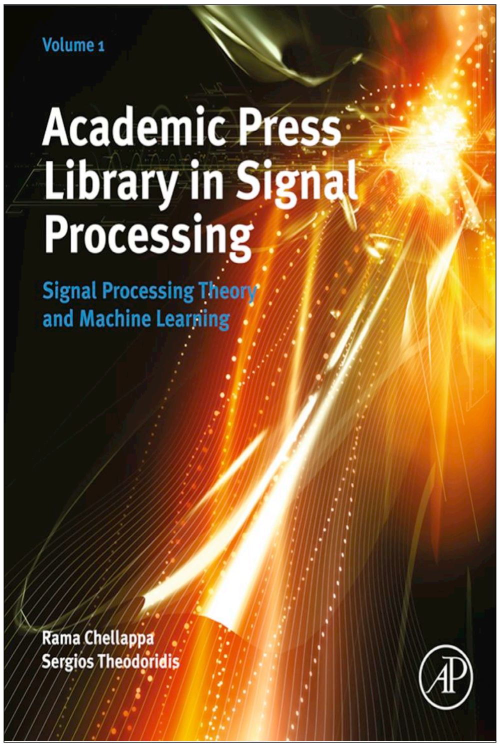
Volume 1
Academic Press Library in Signal Processing
Signal Processing Theory and Machine Learning
Rama Chellappa
Sergios Theodoridis

From Konstantinos Slavakis et al., Online Learning in Reproducing Kernel Hilbert Spaces. In: Rama Chellappa, Sergios Theodoridis, editors, Academic Press Library in Signal Processing. Vol 1, Signal Processing Theory and Machine Learning, Chennai: Academic Press, 2014, p. 883-987.
ISBN: 978-0-12-396502-8
© Copyright 2014 Elsevier Ltd
Academic Press.

CHAPTER 17

Online Learning in Reproducing Kernel Hilbert Spaces

Konstantinos Slavakis*, Pantelis Bouboulis†, and Sergios Theodoridis†

**University of Peloponnese, Department of Telecommunications Science and Technology, Karaiskaki St., Tripolis 22100, Greece*

†University of Athens, Department of Informatics and Telecommunications, Ilissia, Athens 15784, Greece

Nomenclature

Training Data	A set of input-output measurements that are used to train a specific learning machine
Online Learning	A learning task, where the data points arrive sequentially and used only once
Batch Learning	A learning task, where the training data are known beforehand
Regression Task	A learning task that estimates a parametric input-output relationship that best fits the available data
Classification Task	A learning task that classifies the data into one or more classes
Overfitting	A common notion in the Machine Learning literature. If the adopted model is too complex then the learning algorithm tends to fit to the training data as much as possible (e.g., it fits even the noisy data) and cannot perform well when new data arrive
Regularization	A technique that biases the solution towards a smoother result. This is usually achieved by adding the norm of the solution to the cost function of the minimization task
Sparsification	A method that biases the solution towards a result, that the least significant components are pushed to zero
Reproducing Kernel Hilbert Spaces (RKHS)	An inner product space of functions that is complete and has a special structure
Kernel	A positive definite function of two variables that is associated to a specific RKHS

Fréchet Differentiation	A mathematical framework that generalizes the notion of the standard derivative, to more general function spaces
Least Mean Squares	This is a class of stochastic online learning tasks, which aim to find the parameters of an input-output relation by minimizing the least mean squares of the error at each iteration (difference between the desired and the actual output), using a gradient descent rationale
Recursive Least Squares	A class of deterministic online learning tasks, which aim to find the parameters of an input-output relation by minimizing the least mean squares of the total error (the sum of differences between the desired and the actual output up to the current iteration) using a recursive Newton-like method
Adaptive Projected Subgradient Method (APSM)	A convex analytic framework for online learning. Based on set theoretic estimation arguments, and fixed point theory, the APSM is tailored to attack online learning problems related to general, convexly constrained, non-smooth optimization tasks
Wirtinger's Calculus	A methodology for computing gradients of real valued functions that are defined on complex domains, using the standard rules of differentiation

1.17.1 Introduction

A large class of Machine Learning tasks becomes equivalent with estimating an unknown functional dependence that relates the so called input (cause) to the output (effect) variables. Most often, this function is chosen to be of a specific type, e.g., linear, quadratic, etc., and it is described by a set of unknown parameters. There are two major paths in dealing with this set of parameters.

The first one, known as *Bayesian Inference*, treats the parameters as random variables, [1]. According to this approach, the dependence between the input and output variables is expressed via a set of pdfs. The task of learning is to obtain estimates of the pdf that relates the input-output variables as well as the pdf that describes the random nature of the parameters. However, the main concern of the Bayesian Inference approach is not in obtaining specific values for the parameters. Bayesian Inference can live without this information. In contrast, the alternative path, which will be our focus in this paper, considers the unknown parameters to be deterministic and its major concern is to obtain specific estimates for their values.

Both trends share a common trait; they dig out the required information, in order to build up their estimates, from an available set of input-output measurements known as the *training set*. The stage of estimating the pdfs/parameters is known as training and this philosophy of learning is known as *supervised learning*. This is not the only approach which Machine Learning embraces in order to "learn" from data. *Unsupervised* and *semi-supervised* learning are also major learning directions, [2]. In the former, no measurements of the output variable(s) are available in the training data set. In the latter, there are only a few. In this paper, we will focus on the supervised learning of a set of deterministically treated parameters. We will refer to this task as *parameter estimation*.

In the parameter estimation task, depending on the adopted functional dependence, different models result. At this point, it has to be emphasized that the only "truth" that the designer has at his/her disposal is the set of training data. The model that the designer attempts to "fit" in the data is just a concept in an effort to be able to exploit and use the available measurements for his/her own benefit; that is, to be able to make useful predictions of the output values in the future, when new measurements of the inputs, outside the training set, become available. The ultimate goal of the obtained predictions is to assist the designer to make decisions, once the input measurements are disclosed to him/her. In nature, one can never know if there is a "true" model associated with the data. Our confidence in "true" models only concerns simulated data. In real life, one is obliged to adopt a model and a training method to learn its parameters so that the resulting predictions can be useful in practice. The designed Machine Learning system can be used till it is substituted by another, which will be based on a new model and/or a new training method, that leads to improved predictions and which can be computed with affordable computational resources.

In this paper, we are concerned with the more general case of nonlinear models. Obviously, different nonlinearities provide different models. Our approach will be in treating a large class of nonlinearities, which are usually met in practice, in a unifying way by mobilizing the powerful tool of Reproducing Kernel Hilbert Spaces (RKHS). This is an old concept, that was gradually developed within the functional analysis discipline [3] and it was, more recently, popularized in Machine Learning in the context of *Support Vector Machines*. This methodology allows one to treat a wide class of different nonlinearities in a unifying way, by solving an equivalent linear task in a space different than the one where the original measurements lie. This is done by adopting an implicit mapping from the input space to the so called *feature* space. In the realm of RKHSs, one needs not to bother about the specific nature of this space, not even about its dimensionality. Once the parametric training has been completed, the designer can "try" different implicit mappings, which correspond to different nonlinearities and keep the one that fits best his/her needs. This is achieved during the so-called *validation* stage. Ideally, this can be done by testing the performance of the resulting estimator using a data set that is independent of the one used for training. In case this is not possible, then various techniques are available in order to exploit a single data set, both for training and validation, [2].

Concerning the estimation process, once a parametric modeling has been adopted, the parameter estimation task relies on adopting, also, a criterion that measures the quality of fit between the predicted values and the measured ones, over the training set. Different criteria result in different estimates and it is up to the designer to finally adopt the one that better fits his/her needs. In the final step of parameter estimation, an algorithm has to be chosen to optimize the criterion. There are two major philosophies that one has to comply with. The more classical one is the so-called *batch processing* approach. The training data are available in "one go." The designer has access to all of them simultaneously, which are then used for the optimization. In some scenarios, algorithms are developed so that to consider one training point at a time and consider all of them sequentially, and this process goes on till the algorithm converges. Sometimes, such schemes are referred to as online; however, these schemes retain their batch processing flavor, in the sense that the training data set is *fixed* and the number of the points in it is known before the training starts. Hence, we will refer to these as batch techniques and keep the term online for something different.

All batch schemes suffer from a major disadvantage. Once a new measurement becomes available, after the training has been completed, the whole training process has to start from scratch, with a new training set that has been enriched by the extra measurement. No doubt, this is an inefficient way to attack the task. The most efficient way is to have methods that perform the optimization recursively, in the sense of updating the current estimate every time a new measurement is received, by considering *only* this newly received information and the currently available estimate of the parameters. The previously used training data will *never be needed again*. They have become part of the history; all information that they had to contribute to the training process now resides in the current estimate. We will refer to such schemes as *online* or *adaptive*, although many authors call them *time-recursive or sequential*. The term Adaptive Learning is mainly used in Signal Processing and such schemes have also been used extensively in Communications, System Theory and Automatic Control from the early sixties, in the context of the LMS, RLS and Kalman Filtering, e.g., [4–6]. The driving force behind the need for such schemes was the need for the training mechanism to be able to accommodate slow time variations of the learning environment and slowly forget the past. As a matter of fact, such a philosophy imitates better the way that human brain learns and tries to adapt to new situations. Online learning mechanisms have recently become very popular in applications such as Data Mining and Bioinformatics and more general in cases where data reside in large data bases, with massive number of training points, which cannot be stored in the memory and have to be considered one at a time. It is this path of online parameter estimation learning that we will pursue in this paper. As we will see, it turns out that very efficient schemes, of even linear complexity per iteration with respect to the number of the unknown parameters, are now available.

Our goal in this overview is to make an effort to collect a number of online/adaptive algorithms, which have been proposed over a number of decades, in a systematic way under a common umbrella; they are schemes that basically stem from only a few generic recurrences. It is interesting to note that some of these algorithms have been discovered and used under different names in different research communities, without one being referring to the other.

In an overview paper, there is always the dilemma how deep one can go in proofs. Sometimes proofs are completely omitted. We decided to keep some proofs, associated with the generic schemes; those which do not become too technical. Proofs give the newcomer to the field the flavor of the required mathematical tools and also give him/her the feeling of what is behind the performance of an algorithm; in simple words, why the algorithm works. For those who are not interested in proofs, simply, they can bypass them. It was for the less mathematically inclined readers, that we decided to describe first two structurally simple algorithms, the LMS and the RLS. We left the rest, which are derived via convex analytic arguments, for the later stage. As we move on, the treatment may look more mathematically demanding, although we made an effort to simplify things as much as we could. After all, talking about algorithms one cannot just use "words" and a "picture" of an algorithm. Although sometimes this is a line that is followed, the authors of this paper belong to a different school. Thus, this paper may not address the needs of a black-box user of the algorithms.