12-2016

Incremental and Decremental SVM for Regression

Honorius Gâlmeanu

Lucian Mircea Sasu

Răzvan Andonie

INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
ISSN 1841-9836, 11(6):755-775, December 2016.

Incremental and Decremental SVM for Regression

H. Gâlmeanu, L.M. Sasu, R. Andonie

Honorius Gâlmeanu*
1. Siemens Corporate Technology
honorius.galmeanu@siemens.com
2. Faculty of Mathematics and Informatics
Transilvania University of Braşov
*Corresponding author: galmeanu@unitbv.ro

Lucian Mircea Sasu
1. Faculty of Mathematics and Informatics
Transilvania University of Braşov
lmsasu@unitbv.ro
2. Siemens Corporate Technology
lucian.sasu@siemens.com

Răzvan Andonie
1. Computer Science Department
Central Washington University, Ellensburg, USA
andonie@cwu.edu
2. Electronics and Computers Department
Transilvania University of Braşov

Abstract: Training a support vector machine (SVM) for regression (function approximation) in an incremental/decremental way consists essentially in migrating the input vectors in and out of the support vector set with specific modification of the associated thresholds. We introduce with full details such a method, which allows for defining the exact increments or decrements associated with the thresholds before vector migrations take place. Two delicate issues are especially addressed: the variation of the regularization parameter (for tuning the model performance) and the extreme situations where the support vector set becomes empty. We experimentally compare our method with several regression methods: the multilayer perceptron, two standard SVM implementations, and two models based on adaptive resonance theory.
Keywords: support vector machine, incremental and decremental learning, regression, function approximation

1 Introduction

The approximation of continuous functions that are known only at a certain number of discrete points (also known as regression), a standard procedure in statistics, can be also approached from a machine learning perspective. In the context of supervised training, incremental learning means learning each input-output sample pair, without keeping it for subsequent processing. The topic addressed in this paper is supervised incremental/decremental learning of regression models, in particular SVM models. The fundamental issues in incremental/decremental learning are: *i)* how can a learning system adapt to new information without corrupting or forgetting previously learned information (the *stability-plasticity* dilemma addressed by Carpenter and Grossberg [2]), and *ii)* how can a learning system "forget" information without corrupting previously learned information and without adding new information. In other words, how can a system *learn without unlearning* and *unlearn without learning.*

Support Vector Regressions (SVRs) are SVM learning models used for regression. These algorithms solve the quadratic optimization problem using a decomposition method based on Sequential Minimum Optimization [4], or on an incremental learning method. The incremental method considers that input patterns are added to the solution (or removed from the solution), one at a time, without affecting the learning process of the other patterns [13, 14].

After introducing some basic notations, we aim review in this introductory section related work on incremental/decremental SVRs and also highlight our contribution.

1.1 SVR notations

For a non-linear SVR, regression translates into the following optimization problem [14]:

Approximate a given set of training pairs (x_i, y_i), $i = 1 \dots N$, $x_i \in \mathbb{R}^m$, $y_i \in \mathbb{R}$, with an SVM defined by

$$g(x_i) = \mathbf{w}^T \Phi(x_i) + w_0, \tag{1}$$

where $\Phi(\cdot)$ is a non-linear function and w_0 is an offset. The objective function $g(x) : \mathbb{R}^m \to \mathbb{R}$ requires the minimization [21] of

$$\min_{\mathbf{w}, w_0, \xi_i, \xi_i^*} J(\mathbf{w}) = \min_{\mathbf{w}, w_0, \xi_i, \xi_i^*} \left(\frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^{N} (\xi_i + \xi_i^*) \right), \tag{2}$$

and the constraints are given by

$$-\epsilon - \xi_i^* \leq g(x_i) - y_i \leq \epsilon + \xi_i \qquad \text{for } \xi_i, \xi_i^* \geq 0,\ i = 1 \dots N \tag{3}$$

SVR matches the output value of the objective function $g(x_i)$ as close as possible to the given y_i. The output value is constrained to lay inside a "tube" of values, given by $-\epsilon + y_i \leq g(x_i) \leq \epsilon + y_i$, where ϵ is a given constant. The slack variables ξ_i and ξ_i^* allow the system to cope with patterns that do not fall inside the "tube" and also to minimize the acceptable error for such patterns.

To minimize the expression of $J(\mathbf{w})$ in equation (2) with respect to $\mathbf{w}$ and w_0, after several manipulations, the Lagrangian can be written as:

$$L(\mathbf{w}, w_0, \alpha_i, \alpha_i^*) = \\ -\frac{1}{2} \sum_i \sum_j (\alpha_i^* - \alpha_i) \Phi(x_i)^T \Phi(x_j) (\alpha_j^* - \alpha_j) + \sum_i (\alpha_i^* - \alpha_i) y_i - \sum_i (\alpha_i^* - \alpha_i) w_0 - \sum_i (\alpha_i + \alpha_i^*) \epsilon \tag{4}$$

subject to $\alpha_i, \alpha_i^* \geq 0$. The Karush-Kuhn-Tucker (KKT) conditions (imposed by minimizing the Lagrangian) are given by:

$$\mathbf{w} = \sum_i (\alpha_i^* - \alpha_i) \Phi(x_i) \tag{5}$$

$$\sum_i (\alpha_i - \alpha_i^*) = 0 \qquad 0 \leq \alpha_i \leq C \qquad 0 \leq \alpha_i^* \leq C \tag{6}$$

The Wolfe dual form states that this minimization with respect to $\mathbf{w}$ and w_0 can be transformed into a maximization with respect to α_i and α_i^*. This leads to the minimization of $L'(\mathbf{w}, w_0, \alpha_i, \alpha_i^*) = -L(\mathbf{w}, w_0, \alpha_i, \alpha_i^*)$.

The expression of $L'(\mathbf{w}, w_0, \alpha_i, \alpha_i^*)$ depends only on the inner product $\Phi(x_i)^T\Phi(x_j)$. Therefore, we use the notation $Q_{ij} = \Phi(x_i)^T\Phi(x_j)$. However, SVM uses the "kernel trick": the inner products of the vectors $\Phi(x)$, belonging to an infinite-dimension feature space, are replaced by the non-linear kernel function $K(x_i, x_j) = e^{-\sigma(x_i - x_j)^2} = \Phi(x_i)^T\Phi(x_j)$ [19]. Generally, kernel functions are used for non-linear SVMs, so we use the notation $Q_{ij} = K(x_i, x_j)$ to designate the kernel function.

The extremum conditions for $L'(\mathbf{w}, w_0, \alpha_i, \alpha_i^*)$, considering also the conditions (5, 6), are given by:

$$\frac{\partial L'}{\partial \alpha_i} = -\sum_j Q_{ij}(\alpha_j^* - \alpha_j) - w_0 + y_i + \epsilon - \delta_i + u_i = 0 \tag{7}$$

$$\frac{\partial L'}{\partial \alpha_i^*} = \sum_j Q_{ij}(\alpha_j^* - \alpha_j) + w_0 - y_i + \epsilon - \delta_i^* + u_i^* = 0 \tag{8}$$

with the following additional KKT conditions:

$$\delta_i, \delta_i^* \geq 0 \qquad \delta_i\alpha_i = 0, \quad \delta_i^*\alpha_i^* = 0 \tag{9}$$

$$u_i, u_i^* \geq 0 \qquad u_i(C - \alpha_i) = 0, \quad u_i^*(C - \alpha_i^*) = 0 \tag{10}$$

From equation (1), using the condition (5) and the notation $\alpha_j^* - \alpha_j = \theta_j$, the expression of the "margin" function h_i becomes:

$$h_i = g(x_i) - y_i = \sum_j Q_{ij}\theta_j + w_0 - y_i \tag{11}$$

$$\text{where} \qquad -C \leq \theta_i \leq C \tag{12}$$

From equations (7) - (12), observing in the KKT conditions that at most one of the α_i, α_i^* values is nonzero, it follows that there are three possible vector sets:

- *error vectors* (E), for the following cases:

  $h_i < -\epsilon$ (on $\theta_i = C$), or $h_i > \epsilon$ (on $\theta_i = -C$);

- *support vectors* (S), for the cases:

  $h_i = -\epsilon$ (on $0 < \theta_i < C$), or $h_i = \epsilon$ (on $-C < \theta_i < 0$);

- *other vectors* (O), for the cases where

  $-\epsilon < h_i < \epsilon$, with $\theta_i = 0$.

SVMs can be trained in an incremental/decremental way. Using equation (11), we can decompose h_i into two parts: the contribution of other vectors and the contribution of its own vector:

$$h_i = \sum_{j \neq i} Q_{ij}\theta_j + Q_{ii}\theta_i + w_0 - y_i \tag{13}$$

Given that $Q_{ii} > 0$, an increasing negative contribution of other vectors θ_j (for $j \neq i$) would be compensated by increasing the value for θ_i up to the point where $\theta_i = C$. From this point onward, since the increasing of negative contribution for θ_j would no longer be counteracted by

the increase of θ_i, the net result is that h_i would decrease past the $-\epsilon$ value, and x_i would be treated as an error vector.

In equation (13), only the error and support vectors contribute to the expression h_i. For all the sets, the associated threshold value θ_i is bounded by the *regularization parameter* C (equation (12)). $R = \{E \cup O\}$ is the set of *reserve vectors*. We denote by r, s, and e the reserve, support and error vectors, respectively. The whole training set is defined as $\{E \cup S \cup O\}$.

1.2 Related work and our contribution

The *adiabatic* incremental/decremental training algorithm for SVMs (and SVRs) was introduced in [13, 14] and it follows from a method proposed by Cauwenberghs and Poggio [3, 5]. For each pattern being part of the solution (called *support vector*), the associated threshold value is $\theta_i \in \{-\epsilon, \epsilon\}$. Following equation (13), as one new vector is "learned", its associated threshold θ_i increases/decreases, starting from 0, on the expense of other vectors' thresholds. During this process, the rest of the vectors migrate between the sets of support, error and other vectors. When the same vector is "unlearned", its corresponding threshold value θ_i is decreased (when positive) or increased (when negative), in order to reach 0 (when the vector is removed). During unlearning a vector, the vectors which previously migrated between sets when learning that vector will now migrate in the reverse order. An implementation of the adiabatic algorithm can be found in [12], with further developments in [7].

During incremental training, while changing the threshold of the newly introduced vector, the vectors will migrate among sets. The value of the threshold increment determines how many vectors will migrate. Therefore, we are interested to determine the largest threshold increment $\Delta\theta_i$ which is small enough to cause only one vector to migrate. Proceeding with fixed small increments of $\Delta\theta_i$ is not an efficient search technique. We discussed in [7] the conditions for vector migrations between the sets and how to determine the optimal increment. We gave the incremental/decremental context, but we did not describe particular situations, like the empty support vector set situation and the variation of the regularization parameter.

An incremental/decremental SVR, which can simultaneously add batches of new samples and also remove the obsolete ones, was applied for online time series prediction [11]. The ν-SVR, presented in [21], is an incremental/decremental SVR which determines a priori the proportion ν of support vectors from the total set. The algorithms in [9, 11] are variations of the adiabatic algorithm.

For a classification problem, an incremental/decremental SVM operates in the following way: To add vector x_c to the solution, the associated threshold θ_c is set from 0 to C, whereas to remove x_c, θ_c is set to 0. For SVRs, this operations are not symmetric anymore: To add x_c, θ_c is set from 0 to C (increase) or to $-C$ (decrease); to remove x_c, θ_c is also increased or decreased. The direction (incremental or decremental) is not directly given by the intention - adding or removing the vector, but by the margin h_c. We will discuss this aspect in Section 2.

We name our method Incremental/Decremental Support Vector Machine Regression (IDSVMR). It is based on the adiabatic training algorithm. Compared to [13, 14], the IDSVMR contains all aspects needed for an implementation: *a)* We provide the relations that give the exact amount needed for increasing the threshold θ_c for the new vector, before any vector migrates; *b)* We determine the exact threshold variation that leads from one migration to the next. This produces a robust incremental algorithm, that creates a sequence of vector migrations, as part of the learning process; *c)* We deduct the expressions for the variation of the threshold θ_c, from one migration to the next. During training, the support vectors' set may become empty. We give the exact expressions needed to increase θ_c also considering this particular case.

Compared to [7], we contribute with: *i)* a complete description of the increment/decrement

operations of the regularization parameter, showing that the expressions to be maximized (minimized) are similar, and *ii)* we expand the results from [12] by giving an exact procedure on how to continue the migration in the empty support vector set cases, for both the regular procedure and the increment/decrement of the regularization parameter.

Section 2 describes the expressions of the threshold variables θ_i and the expression of the vectors margins before the first migration. Section 3 introduces the expressions of the maximum threshold variations that occur before vectors migrations. Section 4 analyzes the maximum variation of the regularization parameter. Section 5 discusses the variations of the free parameter w_0 and the regularization parameter in case of an empty support vector set. Section 6 presents the results of the IDSVMR implemented as a Weka [10] plugin, compared with the multilayer perceptron (MLP), two models based on adaptive resonance theory, and two classical SVM–derived regression models. Finally, Section 6 contains our conclusions.

2 Incremental and decremental updates of thresholds

We start this section with summarizing the adiabatic training algorithm, which is at the core of the IDSVMR method. Then, we will introduce IDSVMR implementation details which are not included in the original adiabatic algorithm.

To adapt to a new vector x_c, the adiabatic algorithm migrates specific vectors between the sets, in order to reestablish the KKT conditions [5]. At the beginning, x_c has the threshold $\theta_c = 0$. The threshold can evolve both toward positive or negative directions. First, this direction is determined. Then, the threshold is modified considering, with each update, the migration between the sets. The migration is identified by observing the maximum (or minimum) variation of the margins h_i for each vector. Considering the KKT conditions, specifically the conditions for the thresholds (6) and margin function (11), their variations can be expressed as:

$$\Delta h_i = \sum_j Q_{ij} \Delta\theta_j + \Delta w_0 \tag{14}$$

$$0 = \sum_j \Delta\theta_j \tag{15}$$

When adding vector x_c to the set, the migrations that take place strive to keep KKT conditions in place. Thus, the variations of thresholds and margins can be further expanded to:

$$\Delta h_i = \sum_j Q_{ij} \Delta\theta_j + \Delta w_0 + Q_{ic} \Delta\theta_c \tag{16}$$

$$0 = \sum_j \Delta\theta_j + \Delta\theta_c \tag{17}$$

These relations can be written more compact using $\mathbf{e}$, the vector of ones:

$$\begin{bmatrix} \Delta h_S \\ \Delta h_R \\ \Delta h_c \\ 0 \end{bmatrix} = \begin{bmatrix} \mathbf{e}_S & Q_{SS} \\ \mathbf{e}_R & Q_{RS} \\ 1 & Q_{cS} \\ 0 & \mathbf{e}_S^T \end{bmatrix} \begin{bmatrix} \Delta w_0 \\ \Delta\theta_S \end{bmatrix} + \Delta\theta_c \begin{bmatrix} Q_{Sc} \\ Q_{Rc} \\ Q_{cc} \\ 1 \end{bmatrix} \tag{18}$$

The modification of $\Delta\theta_c$ would be absorbed by the modification of $\Delta\theta_S$ (the set of support vectors), Δw_0, and the variation of margin functions. The margins of support vectors do not change: they are either $-\epsilon$ or $+\epsilon$. Hence, we have $\Delta h_S = 0$ and we obtain:

$$\begin{bmatrix} \Delta w_0 \\ \Delta\theta_S \end{bmatrix} = \underbrace{-\begin{bmatrix} 0 & \mathbf{e}_S^T \\ \mathbf{e}_S & Q_{SS} \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ Q_{Sc} \end{bmatrix}}_{\beta} \Delta\theta_c \tag{19}$$

Considering the definitions of Δw_0 and $\Delta\theta_S$, the margin functions of the current and reserve vectors are:

$$\begin{bmatrix} \Delta h_c \\ \Delta h_R \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & Q_{cS} \\ \mathbf{e}_R & Q_{RS} \end{bmatrix} \beta + \begin{bmatrix} Q_{cc} \\ Q_{Rc} \end{bmatrix}}_{\gamma} \Delta\theta_c \tag{20}$$

To prevent the recalculation of the inverse matrix in equation (19), the Sherman-Morrison-Woodbury formula was used in [12]. If used repeatedly, this leads to error accumulation. Therefore, we prefer to calculate the inverse by LU decomposition, which is in $O(n^3)$ but numerically stable [8].

3 Learning direction and extremum increment/decrement before vector migration

In what follows, we describe the conditions for vector migration between the sets and establish migration conditions for all possible situations. We explain how we prevent immediate cycling, mentioned in [12]. Although laborious, we give full coverage of all particular situations.

To "learn" a new vector x_c, the training process starts with the initial value $\theta_c = 0$. Its margin, h_c, is first determined using relation (11). Since $\Delta h_c = \gamma_c \Delta\theta_c$, the aim is to modify θ_c such that the margin function is updated to fit as well as possible into the "tube" according to relation (3). The following cases occur:

1. if $h_c > \epsilon$, then the margin decreases to fit into the $-\epsilon < h_c < \epsilon$ "tube" and $\Delta h_c < 0$:
   - if $\gamma_c > 0$, then $\Delta\theta_c < 0$, θ_c decreases to $-C$, and the direction is **decremental**;
   - else, if $\gamma_c < 0$, then $\Delta\theta_c > 0$, θ_c increases to C, and the direction is **incremental**;
2. if $h_c < -\epsilon$, then the margin increases and $\Delta h_c > 0$:
   - if $\gamma_c > 0$, then $\Delta\theta_c > 0$, θ_c increases to C and the direction is **incremental**;
   - else, if $\gamma_c < 0$, then $\Delta\theta_c < 0$, θ_c decreases to $-C$ and the direction is **decremental**;

From these relations we can establish the direction:

- if $sgn(h_c) \neq sgn(\gamma_c)$, the approach is **incremental** and $\Delta\theta_c > 0$;
- if $sgn(h_c) = sgn(\gamma_c)$, the approach is **decremental** and $\Delta\theta_c < 0$.

To "unlearn" a vector x_c from the support or error sets, the direction only depends on the value of θ_c: if $\theta_c < 0$, the direction is **incremental**; if $\theta_c > 0$, the direction is **decremental**.

Once the direction (the variation of θ_c) is established, we have to determine the variation increment $\Delta\theta_c$. We have the following cases (details are provided in Appendix A):

1. Migration of support (S) vectors.

   Considering only the $\Delta\theta_S$ components from equation (19) and following the β notation, the expression of the increment for only one support vector is $\Delta\theta_s = \beta_s \cdot \Delta\theta_c$. It gives the limits for the θ_s updates (we considered β_s to be the specific component from the β vector). On the other hand, considering the allowed variation interval for θ_s, we can determine the limit for θ_c before the support vector associated with θ_s leaves the set:

   - the **incremental** case:
     - if $sgn(h_s) = sgn(\beta_s)$, θ_s may reach 0, so we have the migration to set O:

       $$\Delta\theta_c \leq -\frac{\theta_s}{\beta_s} \qquad (S \to O)$$

     - if $sgn(h_s) \neq sgn(\beta_s)$, θ_s may reach C or $-C$:

       $$\Delta\theta_c \leq \frac{sgn(\beta_s) \cdot C - \theta_s}{\beta_s} \qquad (S \to E)$$

   - the **decremental** case:
     - if $sgn(h_s) \neq sgn(\beta_s)$, θ_s may reach 0:

       $$\Delta\theta_c \geq -\frac{\theta_s}{\beta_s} \qquad (S \to O)$$

     - if $sgn(h_s) = sgn(\beta_s)$, θ_s may reach C or $-C$:

       $$\Delta\theta_c \geq \frac{-sgn(\beta_s) \cdot C - \theta_s}{\beta_s} \qquad (S \to E)$$

2. Migration of other (O) vectors. These are the vectors with $-\epsilon < h_e < \epsilon$:

   - for the **incremental** case:

     $$\Delta\theta_c \leq \frac{sgn(\gamma_r) \cdot \epsilon - h_r}{\gamma_r} \qquad (O \to S)$$

   - for the **decremental** case:

     $$\Delta\theta_c \geq \frac{-sgn(\gamma_r) \cdot \epsilon - h_r}{\gamma_r} \qquad (O \to S)$$

3. Migration of error (E) vectors. These are the vectors with $h_e < -\epsilon$ or $h_e > \epsilon$:

   - for the **incremental** case:
     - if $sgn(h_r) \neq sgn(\gamma_r)$:

       $$\Delta\theta_c \leq \frac{-sgn(\gamma_r) \cdot \epsilon - h_r}{\gamma_r} \qquad (E \to S)$$

     - if $sgn(h_r) = sgn(\gamma_r)$: vector does not migrate
   - for the **decremental** case:
     - if $sgn(h_r) = sgn(\gamma_r)$:

       $$\Delta\theta_c \geq \frac{sgn(\gamma_r) \cdot \epsilon - h_r}{\gamma_r} \qquad (E \to S)$$

– if $sgn(h_r) \neq sgn(\gamma_r)$: vector does not migrate

4. Migration of current x_c vector. The current vector x_c newly added to the set is checked for possible migration to the support set (S):

   - for the **incremental** case:
     – if $sgn(h_c) \neq sgn(\gamma_c)$:

$$\Delta\theta_c \leq \frac{-sgn(\gamma_c) \cdot \epsilon - h_c}{\gamma_c} \qquad (E \to S)$$

     – if $sgn(h_r) = sgn(\gamma_c)$: vector does not migrate
   - for the **decremental** case:
     – if $sgn(h_c) = sgn(\gamma_c)$:

$$\Delta\theta_c \geq \frac{sgn(\gamma_c) \cdot \epsilon - h_c}{\gamma_c} \qquad (E \to S)$$

     – if $sgn(h_c) \neq sgn(\gamma_c)$: vector does not migrate

   For removing x_c from the training set, the objective is to increase/decrease θ_c to zero, so that the variation of the margin does not determine vector migration.

5. The variation of θ_c from/to zero should be $-C \leq \theta_c \leq C$, and we have the following conditions:

   - for **adding** x_c:
     – for the **incremental** case, $\Delta\theta_c \leq C - \theta_c$
     – for the **decremental** case, $\Delta\theta_c \geq -C - \theta_c$
   - for **removing** x_c:
     – for the **incremental** case, where $\theta_c < 0$, $\Delta\theta_c \leq -\theta_c$
     – for the **decremental** case, where $\theta_c > 0$, $\Delta\theta_c \geq -\theta_c$

The maximum variation of $\Delta\theta_c$ for the **incremental** case, before the vectors change sets, is determined by computing the minimum of all these values. Conversely, the minimum variation for the **decremental** case is established by determining the maximum of all these (negative) values deduced in the five cases presented above.

4 Incrementing and decrementing the regularization parameter

Training a SVM also requires at some stage a fine-tuning of the regularization parameter. Many times, this is done by trial-and-error. In order to avoid re-training from scratch, the adiabatic algorithm modifies the regularization parameter in an incremental or decremental way, in small increments, watching for vectors migrations between sets. In the following, we aim to determine the maximum values for these increments.

Starting from equations (14) and (15), and adopting the notation $\theta_k = b_k \cdot C$ for the threshold of error vectors, the relations for the margin functions (when only the regularization parameter is modified) can be written as:

$$\Delta h_i = \sum_{j \in S} Q_{ij} \Delta\theta_j + \left(\sum_{k \in E} Q_{ik} b_k \right) \Delta C + w_0 \tag{21}$$

$$0 = \sum_{j \in S} \Delta\theta_j + \left(\sum_{k \in E} b_k \right) \Delta C \tag{22}$$

Or, more compact:

$$\begin{bmatrix} \Delta h_S \\ \Delta h_R \\ 0 \end{bmatrix} = \begin{bmatrix} \mathbf{e}_S & Q_{SS} \\ \mathbf{e}_R & Q_{RS} \\ 0 & \mathbf{e}_S^T \end{bmatrix} \cdot \begin{bmatrix} \Delta w_0 \\ \Delta\theta_S \end{bmatrix} + \begin{bmatrix} \sum_{k \in E} b_k Q_{Sk} \\ \sum_{k \in E} b_k Q_{Rk} \\ \sum_{k \in E} b_k \end{bmatrix} \cdot \Delta C \tag{23}$$

Since the margins for the support vectors remain constant, $\Delta h_S = 0$, the variations of support vectors threshold and the margin variations can be computed as

$$\begin{bmatrix} \Delta w_0 \\ \Delta\theta_S \end{bmatrix} = \underbrace{- \begin{bmatrix} 0 & \mathbf{e}_S^T \\ \mathbf{e}_S & Q_{SS} \end{bmatrix}^{-1} \cdot \underbrace{\begin{bmatrix} \sum_{k \in E} b_k \\ \sum_{k \in E} b_k Q_{Sk} \end{bmatrix}}_{\eta}}_{\beta} \Delta C \tag{24}$$

$$\Delta h_R = \underbrace{\left(\begin{bmatrix} \mathbf{e}_R & Q_{RS} \end{bmatrix} \beta + \sum_{k \in E} b_k Q_{Rk} \right)}_{\gamma} \Delta C \tag{25}$$

Let us summarize the migration conditions for each vector type (details can be found in Appendix B).

1. Migration of support vectors.

   - for **incremental** ($\Delta C > 0$, cumulative conditions):
     - if $sgn(\beta_s + sgn(h_s)) \neq sgn(h_s)$, then the limit is imposed by (migration $S \to E$):

$$\Delta C \leq \frac{-sgn(h_s) \cdot C - \theta_s}{\beta_s + sgn(h_s)}$$

     - if $sgn(\beta_s) = sgn(h_s)$, then the limit is imposed by (migration $S \to O$):

$$\Delta C \leq -\frac{\theta_s}{\beta_s}$$

   - for **decremental** ($\Delta C < 0$, cumulative):
     - if $sgn(\beta_s + sgn(h_s)) = sgn(h_s)$, then the limit is imposed by (migration $S \to E$):

$$\Delta C \geq \frac{-sgn(h_s) \cdot C - \theta_s}{\beta_s + sgn(h_s)}$$

     - if $sgn(\beta_s) \neq sgn(h_s)$, then the limit is imposed by (migration $S \to O$):

$$\Delta C \geq -\frac{\theta_s}{\beta_s}$$

2. Migration of other vectors.

   For other (O) vectors, $-\epsilon < h_r < \epsilon$:

- for **incremental** ($\Delta C > 0$):

$$\Delta C \leq \frac{sgn(\gamma_r)\epsilon - h_r}{\gamma_r}$$

- for **decremental** ($\Delta C < 0$):

$$\Delta C \geq \frac{-sgn(\gamma_r)\epsilon - h_r}{\gamma_r}$$

3. Migration of error (E) vectors.

- for **incremental** ($\Delta C > 0$):
  (a) if $sgn(h_r) \neq sgn(\gamma_r)$:

$$\Delta C \leq \frac{-sgn(\gamma_r)\epsilon - h_r}{\gamma_r}$$

  (b) if $sgn(h_r) = sgn(\gamma_r)$, the limit is not active;
- for **decremental** ($\Delta C < 0$):
  (a) if $sgn(h_r) = sgn(\gamma_r)$:

$$\Delta C \geq \frac{sgn(\gamma_r)\epsilon - h_r}{\gamma_r}$$

  (b) if $sgn(h_r) \neq sgn(\gamma_r)$, the limit is not active;

4. Variation of C is also limited by a target value C_{target}:

- for **incremental**:

$$\Delta C \leq C_{target} - C$$

- for **decremental**:

$$\Delta C \geq C_{target} - C$$

When varying C, the maximum variation before the vectors change sets is given by the minimum of these previously described ΔC values for the **incremental** scenario, or as the maximum, for the **decremental** case.

5 Migration when the support vector set is empty

During training process, the support vector set may become empty. In this circumstance, the variations of the free parameter w_0 and the regularization parameter are very particular.

5.1 Regular training when the support vector set is empty

For this case, equations (19) and (20) cannot be used to determine new values for the threshold parameters. Equation (16) can be written considering that there are no support vectors and that the threshold values for the error and other vectors do not change:

$$\Delta h_c = \Delta h_r = \Delta w_0 \tag{26}$$

The variation of the margin for the current vector is the same as the variation for the margins of all vectors. Imposing limits on the allowed change in margins, before some vector change sets, will lead to determine the specific vector that migrates first. Depending on desired direction (increase or decrease of θ_c), we have two cases:

1. if $h_c < -\epsilon$, then $\theta_c > 0$ increases, we have the following limits for margins:
   - for other vector x_o, $-\epsilon < h_o < \epsilon$, $\Delta h_o \leq \epsilon - h_o$;
   - for error vector x_e with $h_e > \epsilon$, the restriction is not active;
   - for error vector x_e with $h_e < -\epsilon$, $\Delta h_e \leq -\epsilon - h_e$;
   - for x_c, $\Delta h_c \leq -\epsilon - h_c$.
2. if $h_c > \epsilon$, then $\theta_c < 0$ decreases, we have the following limits for margins:
   - for other vector x_o, $-\epsilon < h_o < \epsilon$, $\Delta h_o \geq -\epsilon - h_o$;
   - for error vector x_e with $h_e > \epsilon$, $\Delta h_e \geq \epsilon - h_e$;
   - for error vector x_e with $h_e < -\epsilon$, the restriction is not active;
   - for x_c, $\Delta h_c \geq \epsilon - h_c$.

The rules are valid for either the incremental or the decremental case. The margin of the x_c vector is sought to either increase (first situation) or decrease (second).

It is sufficient to take Δw_0 to be the smallest (or largest, depending on the desired direction) of these quantities. This way of varying the margins would always guarantee that some of them would be equal to $-\epsilon$ or ϵ limit, ensuring that the support vector set would not become empty.

5.2 Updating the regularization parameter when the support vector set is empty

When the support vector set is empty, the regularization parameter cannot modify on the expense of the variation of support vectors' threshold values. Relations (21) and (22) will change to:

$$\Delta h_i = \left(\sum_{k \in E} Q_{ik} b_k \right) \Delta C + \Delta w_0 \tag{27}$$

$$0 = \sum_{k \in E} b_k \Delta C \tag{28}$$

Relation (28) is always valid, regardless of C's variation. As opposed to how we proceeded previously, we assume that w_0 parameter is kept unchanged, thus $\Delta h_i = \mu_i \Delta C$, where we made the notation $\mu_i = \sum_{k \in E} Q_{ik} b_k$.

1. when C increases ($\Delta C > 0$, incremental):
   (a) for other vectors $x_r \in O$, $-\epsilon < h_r < \epsilon$:
      - if $\mu_r > 0$ then $\Delta h_r > 0$, thus $\Delta h_r \leq \epsilon - h_r$ or $\mu_r \Delta C \leq \epsilon - h_r$, which leads to $\Delta C \leq \frac{\epsilon - h_r}{\mu_r}$;
      - if $\mu_r < 0$ then $\Delta h_r < 0$, thus $\Delta h_r \geq -\epsilon - h_r$ or $\mu_r \Delta C \geq -\epsilon - h_r$, which leads to $\Delta C \leq \frac{-\epsilon - h_r}{\mu_r}$;

   (b) for error vectors $x_r \; inE$ and $\theta_r = C$, $h_r < -\epsilon$:
      - if $\mu_r > 0$ then $\Delta h_r > 0$, thus $\Delta h_r \leq -\epsilon - h_r$ or $\mu_r \Delta C \leq -\epsilon - h_r$, which leads to $\Delta C \leq \frac{-\epsilon - h_r}{\mu_r}$;
      - if $\mu_r > 0$ then $\Delta h_r < 0$, the condition is not active;

   (c) for error vectors $x_r \; inE$ and $\theta_r = -C$, $h_r > \epsilon$:
      - if $\mu_r > 0$ then $\Delta h_r > 0$, the condition is not active;

- if $\mu_r < 0$ then $\Delta h_r < 0$, thus $\Delta h_r \geq \epsilon - h_r$ or $\mu_r \Delta C \geq \epsilon - h_r$, which leads to $\Delta C \leq \frac{\epsilon - h_r}{\mu_r}$;

In brief, a minimum will be computed among all the thresholds computed for vectors, imposed by conditions like:

$$\Delta C \leq \frac{set \cdot sgn(\mu_r)\epsilon - h_r}{\mu_r} \tag{29}$$

where $set = 1$ if $x_r \in O$ and $set = -1$ if $x_r \in E$. The condition will be inactive if $sgn(\mu_r) = sgn(h_r)$ for $x_r \in E$.

2. when C decreases ($\Delta C < 0$, incremental): Proceeding in the same manner, a maximum will be computed among all the thresholds determined by similar conditions:

$$\Delta C \geq \frac{-set \cdot sgn(\mu_r)\epsilon - h_r}{\mu_r} \tag{30}$$

where set is defined like above. The condition will be inactive if $sgn(\mu_r) \neq sgn(h_r)$, for $x_r \in E$.

6 Experimental results

In our experiments, we compare the IDSVMR with the MLP, two SVM–based regression models, and two incremental learning models derived by us in previous work from adaptive resonance theory: the Fuzzy ARTMAP with Relevance (FAMR) and the Bayesian ARTMAP for Regression (BAR). Since the last two models are less known, we included here a short description of them.

FAMR was introduced in [1], as an extension of the classical Fuzzy ARTMAP (FAM) [15] and PROBART [16]. FAMR can be used as classifier, regression model and posterior probability estimator. Each training pattern can come with its own relevance factor assigned to it, which influences the probabilistic links formed between the input and output categories. A relevance factor allows for ranking of sample pairs according to the confidence one has in the information source. FAMR builds one–to–many mappings between input and output categories through maximum likelihood, approximating the probability of associations between input and output categories. FAMR comes with a stochastic approximation procedure, which allows making use of the relevance factor associated to the training patterns. Under some mild constraints, FAMR's Mapfield values converge both in mean square and with probability one to the posterior probability $P(k|j)$ between the jth input category and the kth output category.

Bayesian ARTMAP (BA) [17] is a neural architecture which uses a combination of FAM competitive learning and Bayesian learning. BA uses Gaussian categories and FAM competitive learning. BA modifies some of the characteristics of the FAM algorithm mainly by replacing the hyperrectangular categories with Gaussian categories. The categories may grow or shrink, and they are probabilistically associated to classes. BA probabilistically infers the classes associated to input categories, by using all input categories, unlike the competitive approach used by FAM and FAMR. In [18], the BA is extended for function approximation. The resulted architecture – Bayesian ARTMAP for Regression (BAR) – generalizes the BA algorithm using the clustering functionality of both input and output modules. The BAR has the universal approximation capability and also the best approximation property; this is very important in regression [18]. Both the FAMR and the BAR can be trained incrementally, but not decrementally. This is typical for FAM models.

For benchmarking, we use the following public datasets [6]: CPU Computer Hardware (CPU), Boston Housing (BH), Wisconsin Breast Cancer (WBC), and Communities and Crime (CC). For the CPU datasets we removed the following input features: vendor name, model name, and estimated relative performance. From the WBC dataset, we removed the four instances with missing values and also the outcome attribute. The first five features of the CC dataset are acknowledged (by the dataset donors) not to be predictive. We removed them, together with the features with missing values.

The original datasets are described in [6]. The filtered datasets, as explained above[1], are synthetically described in Table 1. To support performance comparisons across various benchmarks, both input and output values are independently scaled between 0 and 1.

Table 1: Synthetic description of the four benchmark datasets used. Some instances and/or attributes were removed from the original datasets

Dataset	Input attributes	Instances
CPU	6	209
BH	13	506
WBC	32	194
CC	99	1994

We compare the regression capabilities of IDSVMR, BAR, and FAMR. We are also interested in comparing these incremental models with the popular MLP model and two classical SVM-based regression algorithms, namely ε-SVR [20] and ν-SVR [21]. This is of interest because IDSVMR, ε-SVR and ν-SVR have overlapping inductive bias, and beyond that they cover incremental and non-incremental adaptive models. We include MLP in the group of non–incremental models. Even if is trained with mini–batches of data, the MLP does not learn new information without possibly corrupting previously learned information (the stability-plasticity condition is not addressed).

For FAMR, BAR and IDSVMR, we use our own Weka plugins, whereas for ε-SVR, ν-SVR and MLP we use the Weka–provided implementations [10].

We use ten random permutations (shuffles) of each datasets. For each permutation, 66% of the data is used as training/validation set and the rest serves exclusively as test set. A ten–fold cross-validation on the training/validation set is performed, for fine-tuning the hyperparameters of each model. For MLP, BAR and FAMR, paper [18] enumerates the sought hyperparameters and their corresponding search ranges. For IDSVMR, the hyperparameters are: the regularization coefficient C (shown in equation 2) and σ, the width of the Gaussian kernel[2]. For the CC dataset, we seek an optimal value of C in $\{4, 4.5, \ldots, 8\}$, and σ is sought over the candidate values $\{0.5, 0.7, 0.9\}$. For the other three datasets, C is sought in the set $\{0.2, 0.4, \ldots, 10\}$, and the candidate values for σ are in the set $\{0.5, 0.6, \ldots, 0.9\}$. The same hyperparameters and corresponding search ranges as for IDSVMR are used for both ε-SVR and for ν-SVR.

After cross-validation, we train the optimized models on the whole learning/validation set, and their generalization capability is assessed on the test set. The test set is used solely in this final assessment. The reported scoring values are the root mean squared error (RMSE) and the mean absolute error (MAE). Table 2 contains the average RMSE and MAE values over the ten permutations, measured on the test sets. We split the table into two sections: the former contains only incremental models, the latter is devoted to MLP and to the two SVM–based regression models.

[1] available at http://www.liaad.up.pt/~ltorgo/Regression/DataSets.html

[2] Note that Section 1.1 and the subsequent ones "hide" this hyperparameter under the notation Q_{ij}. Nevertheless, one ought to seek for a proper value of it as well.

The results for MLP, FAMR, and BAR are the ones found in [18]. All the steps for cross-validation, learning, and performance assessment are performed under the Weka Experimenter framework [10].

Table 2: The performance assessment results for ten random permutations of each of the four benchmarks datasets. The results are split into two groups: the former refers the incremental models (FAMR, BAR and IDSVMR), the latter presents the scores for ε-SVR, ν-SVR and MLP. Every cell of the table contains the pair of RMSE and MAE values averaged on the test sets. The boldfaced values are optimal in their corresponding group. The loss values for MLP, FAMR and BAR are the ones reported in [18].

Model	CPU	BH	WBC	CC
FAMR	**0.09**, 0.06	0.15, 0.12	0.31, 0.27	0.21, 0.14
BAR	0.12, 0.06	0.17, 0.12	0.27, 0.23	0.20, 0.14
IDSVMR	0.12, **0.05**	**0.09**, **0.05**	**0.26**, **0.22**	**0.14**, **0.09**
ε-SVR	0.07, 0.05	**0.09**, 0.06	**0.26**, **0.22**	0.16, 0.12
ν-SVR	**0.05**, **0.02**	**0.09**, **0.05**	**0.26**, **0.22**	**0.15**, **0.11**
MLP	0.19, 0.14	0.15, 0.11	0.30, 0.25	**0.15**, 0.12

For all four datasets, in the group of incremental models, the minimum MAE is obtained by IDSVMR. Except for the CPU dataset, the lowest RMSE is produced by IDSVMR; for CPU, IDSVMR produces the median RMSE value, at tie with the BAR. For the BH dataset, IDSVMR largely outperforms the other models. For all four datasets, IDSVMR exhibits a consistent tendency to produce the lowest error scores.

In case of non–incremental models, ν-SVR always obtains the lowest scores. Except for the CPU dataset, the ε-SVR and ν-SVR models have a similar performance. MLP's scores are close to the ones obtained by ν-SVR for CC, while for the other ones MLP shows rather modest performances.

Finally, we compare the incremental and non–incremental models. For CPU, ν-SVR obtains better performance scores than any incremental model. For BH and WBC, ν-SVR is at par with IDSVMR. For CC, IDSVMR outperforms any non–incremental model.

Conclusion

The IDSVMR proves to be a performant and functional implementation of the adiabatic incremental/decremental SVR model. We have described it here with complete implementation details and we have implemented it as a Weka plugin.

In conclusion, the newly–introduced IDSVMR shows itself as a promising incremental regression model, favorably comparing with the other three incremental algorithms in terms of performance. Due to the differences between the hyperparameters number and ranges, it is rather meaningless to consider execution time for the cross-validation stage used for hyperparameter optimization.

SVMs (and SVRs) are known be relatively slow during training. Using incremental/decremental training is therefore very attractive when dealing with large datasets and with data streams.

Bibliography

[1] Andonie, R.; Sasu, L. (2006); Fuzzy ARTMAP with input relevances, *IEEE Transactions on Neural Networks*, 17: 929-941.

[2] Carpenter, G.A.; Grossberg, S. (1988); The ART of Adaptive Pattern Recognition by a Self-Organizing Neural Network, *IEEE Computer*, 77–88.

[3] Cauwenberghs, G.; Poggio, T. (2000); Incremental and Decremental Support Vector Machine Learning, *Neural Information Processing Systems*, 409-415.

[4] Chang, C.C.; Lin, C.J. (2001); LIBSVM: a Library for Support Vector Machines, Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm

[5] Diehl, C.P.; Cauwenberghs, G. (2003); SVM Incremental Learning, Adaptation and Optimization, *Proceedings of the IJCNN*, 4: 2685-2690.

[6] Frank, A., Asuncion, A. (2010); UCI Machine Learning Repository, http://archive.ics.uci.edu/ml

[7] Galmeanu, H.; Andonie, R. (2008); Incremental/Decremental SVM for Function Approximation, *Proceedings of the 11th Intl. Conf. on Optimization of Electrical and Electronic Equipment*, 2: 155-160.

[8] Golub, G.H.; Van Loan, C.F. (1996); *Matrix Computations*, JHU Press, Baltimore and London, 1996.

[9] Gu, B.; Wang, J.D.; Yu, Y.; Zheng, G.S.; Yu Fan.; Xu, T. (2012); Accurate on-line v-support vector learning, *Neural Networks*, 27: 51–59.

[10] Hall, M.; Frank, E.; Holmes, G.; Pfahringer, B.; Reutemann, P.; Witten, I. H. (2009); The WEKA data mining software: an update, *SIGKDD Explorations Newsletter*, 11 (1):10–18.

[11] Karasuyama, M.; Takeuchi, I. (2010); Multiple incremental decremental learning of support vector machines, *IEEE Transactions on Neural Networks*, 21(7): 1048–1059.

[12] Laskov, P.; Gehl, C.; Krüger, S.; Müller, K.R. (2006); Incremental Support Vector Learning: Analysis, Implementation and Applications, *Journal of Machine Learning Research*, 7: 1909–1936.

[13] Martin, M. (2002); *On-line Support Vector Machines for Function Approximation*, Technical Report LSI-02-11-R, Software Department, Universitat Politecnica de Catalunya.

[14] Ma, J.; Thelier, J.; Perkins, S. (2003); Accurate On-line Support Vector Regression, *Neural Computation*, 15(11): 2683–2703.

[15] Carpenter, G. A.; Grossberg, S.; Markuzon, N.; Reynolds, J. H.; Rosen, D. B., (1992), Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps, *IEEE Transactions on Neural Networks*, 3: 698-713.

[16] Marriott, S.; Harrison, R. F. (1995), A modified fuzzy ARTMAP architecture for the approximation of noisy mappings, *Neural Networks*, 8(4): 619–641.

[17] Vigdor, B.; Lerner, B. (2007). The Bayesian ARTMAP, *IEEE Transactions on Neural Networks* 18: 1628–1644.

[18] Sasu, L.M.; Andonie, R. (2013); Bayesian ARTMAP for regression, *Neural Networks*, ISSN 0893-6080, 46: 23-31.

[19] Shashua, A. (2009); Introduction to Machine Learning: Class Notes 67577, https://arxiv.org/abs/0904.3664v1

[20] Vapnik, V. (1998); *Statistical learning theory*, New York: Wiley.

[21] Schölkopf, B; Smola, A. J; Williamson, R. C.; Bartlett, P. L. (2000); New support vector algorithms, *Neural computation*, 12 (5): 1207–1245.

Appendix A

Migration of vectors on learning and unlearning

1. Migration of support (S) vectors.

   From equation (19), $\Delta\theta_s = \beta_s \cdot \Delta\theta_c$, the following cases are possible:

   (a) $h_s = \epsilon$, for $-C < \theta_s < 0$:

   i. if **incremental** ($\Delta\theta_c > 0$) and $\beta_s > 0$:
   $\Delta\theta_s > 0$, θ_s can reach 0, $\beta_s \cdot \Delta\theta_c = \Delta\theta_s \leq 0 - \theta_s$, thus $\Delta\theta_c \leq -\frac{\theta_s}{\beta_s}$ $(S \to O)$

   ii. if **incremental** ($\Delta\theta_c > 0$) and $\beta_s < 0$:
   $\Delta\theta_s < 0$, θ_s can reach $-C$, $\beta_s \cdot \Delta\theta_c = \Delta\theta_s \geq -C - \theta_s$, thus $\Delta\theta_c \leq \frac{-C-\theta_s}{\beta_s}$ $(S \to E)$

   iii. if **decremental** ($\Delta\theta_c < 0$) and $\beta_s > 0$:
   $\Delta\theta_s < 0$, θ_s can reach -C, $\beta_s \cdot \Delta\theta_c = \Delta\theta_s \geq -C - \theta_s$, thus $\Delta\theta_c \geq \frac{-C-\theta_s}{\beta_s}$ $(S \to E)$

   iv. if **decremental** ($\Delta\theta_c < 0$) and $\beta_s < 0$:
   $\Delta\theta_s > 0$, θ_s can reach 0, $\beta_s \cdot \Delta\theta_c = \Delta\theta_s \leq 0 - \theta_s$, thus $\Delta\theta_c \geq -\frac{\theta_s}{\beta_s}$ $(S \to O)$

   (b) $h_s = -\epsilon$, for $0 < \theta_s < C$:

   i. if **incremental** ($\Delta\theta_c > 0$) and $\beta_s > 0$:
   $\Delta\theta_s > 0$, θ_s can reach C, $\beta_s \cdot \Delta\theta_c = \Delta\theta_s \leq C - \theta_s$, thus $\Delta\theta_c \leq \frac{C-\theta_s}{\beta_s}$ $(S \to E)$

   ii. if **incremental** ($\Delta\theta_c > 0$) and $\beta_s < 0$:
   $\Delta\theta_s < 0$, θ_s can reach 0, $\beta_s \cdot \Delta\theta_c = \Delta\theta_s \geq 0 - \theta_s$, thus $\Delta\theta_c \leq -\frac{\theta_s}{\beta_s}$ $(S \to O)$

   iii. if **decremental** ($\Delta\theta_c < 0$) and $\beta_s > 0$:
   $\Delta\theta_s < 0$, θ_s can reach 0, $\beta_s \cdot \Delta\theta_c = \Delta\theta_s \geq 0 - \theta_s$, thus $\Delta\theta_c \geq -\frac{\theta_s}{\beta_s}$ $(S \to O)$

   iv. if **decremental** ($\Delta\theta_c < 0$) and $\beta_s < 0$:
   $\Delta\theta_s > 0$, θ_s can reach C, $\beta_s \cdot \Delta\theta_c = \Delta\theta_s \leq C - \theta_s$, thus $\Delta\theta_c \geq \frac{C-\theta_s}{\beta_s}$ $(S \to E)$

2. Migration of other (O) vectors.

   These are the vectors with $-\epsilon < h_e < \epsilon$, $\theta_r = 0$, $\Delta h_r = \gamma_r \cdot \Delta\theta_c$.

   (a) if **incremental** ($\Delta\theta_c > 0$) and $\gamma_r > 0$:
   $\Delta h_r > 0$, h_r can reach ϵ, $\gamma_r \cdot \Delta\theta_c = \Delta h_r \leq \epsilon - h_r$, thus $\Delta\theta_c \leq \frac{\epsilon - h_r}{\gamma_r}$ $(O \to S)$

   (b) if **incremental** ($\Delta\theta_c > 0$) and $\gamma_r < 0$:
   $\Delta h_r < 0$, h_r can reach $-\epsilon$, $\gamma_r \cdot \Delta\theta_c = \Delta h_r \geq -\epsilon - h_r$, thus $\Delta\theta_c \leq \frac{-\epsilon - h_r}{\gamma_r}$ $(O \to S)$

(c) if **decremental** ($\Delta\theta_c < 0$) and $\gamma_r > 0$:
$\Delta h_r < 0$, h_r can reach $-\epsilon$, $\gamma_r \cdot \Delta\theta_c = \Delta h_r \geq -\epsilon - h_r$, thus $\Delta\theta_c \geq \frac{-\epsilon - h_r}{\gamma_r}$ $(O \to S)$

(d) if **decremental** ($\Delta\theta_c < 0$) and $\gamma_r < 0$:
$\Delta h_r > 0$, h_r can reach ϵ, $\gamma_r \cdot \Delta\theta_c = \Delta h_r \leq \epsilon - h_r$, thus $\Delta\theta_c \geq \frac{\epsilon - h_r}{\gamma_r}$ $(O \to S)$

3. Migration of the error (E) vectors.

(a) for $h_r > \epsilon$, $\theta_r = -C$, $\Delta h_r = \gamma_r \cdot \Delta\theta_c$:

i. if **incremental** ($\Delta\theta_c > 0$) and $\gamma_r > 0$:
$\Delta h_r > 0$, h_r increases even further, vector does not migrate;

ii. if **incremental** ($\Delta\theta_c > 0$) and $\gamma_r < 0$:
$\Delta h_r < 0$, h_r may reach ϵ, $\gamma_r \cdot \Delta\theta_c = \Delta h_r \geq \epsilon - h_r$, $\Delta\theta_c \leq \frac{\epsilon - h_r}{\gamma_r}$ $(E \to S)$

iii. if **decremental** ($\Delta\theta_c < 0$) and $\gamma_r < 0$:
$\Delta h_r > 0$, h_r increases even further, vector does not migrate;

iv. if **decremental** ($\Delta\theta_c < 0$) and $\gamma_r > 0$:
$\Delta h_r < 0$, h_r may reach ϵ, $\gamma_r \cdot \Delta\theta_c = \Delta h_r \geq \epsilon - h_r$, $\Delta\theta_c \geq \frac{\epsilon - h_r}{\gamma_r}$ $(E \to S)$

(b) for $h_r < -\epsilon$, $\theta_r = C$, $\Delta h_r = \gamma_r \cdot \Delta\theta_c$:

i. if **incremental** ($\Delta\theta_c > 0$) and $\gamma_r > 0$:
$\Delta h_r > 0$, h_r may reach $-\epsilon$, $\gamma_r \cdot \Delta\theta_c = \Delta h_r \leq -\epsilon - h_r$, $\Delta\theta_c \leq \frac{-\epsilon - h_r}{\gamma_r}$ $(E \to S)$

ii. if **incremental** ($\Delta\theta_c > 0$) and $\gamma_r < 0$:
$\Delta h_r < 0$, h_r decreases even further, vector does not migrate;

iii. if **decremental** ($\Delta\theta_c < 0$) and $\gamma_r > 0$:
$\Delta h_r < 0$, h_r decreases even further, vector does not migrate;

iv. if **decremental** ($\Delta\theta_c < 0$) and $\gamma_r < 0$:
$\Delta h_r > 0$, h_r may reach $-\epsilon$, $\gamma_r \cdot \Delta\theta_c = \Delta h_r \leq -\epsilon - h_r$, $\Delta\theta_c \geq \frac{-\epsilon - h_r}{\gamma_r}$ $(E \to S)$

4. Migration of the current x_c vector. The current newly added vector x_c is checked for possible migration to the support set (S):

(a) for $h_c > \epsilon$:

i. if **incremental** ($\Delta\theta_c > 0$) and $\gamma_c > 0$:
$\Delta h_c > 0$, h_c increases even further, vector does not migrate;

ii. if **incremental** ($\Delta\theta_c > 0$) and $\gamma_c < 0$:
$\Delta h_c < 0$, h_c may reach ϵ, $\gamma_c \cdot \Delta\theta_c = \Delta h_c \geq \epsilon - h_c$, $\Delta\theta_c \leq \frac{\epsilon - h_c}{\gamma_c}$ $(E \to S)$

iii. if **decremental** ($\Delta\theta_c < 0$) and $\gamma_c > 0$:
$\Delta h_c < 0$, h_c may reach ϵ, $\gamma_c \cdot \Delta\theta_c = \Delta h_c \geq \epsilon - h_c$, $\Delta\theta_c \geq \frac{\epsilon - h_c}{\gamma_c}$ $(E \to S)$

iv. if **decremental** ($\Delta\theta_c < 0$) and $\gamma_c < 0$:
$\Delta h_c > 0$, h_c increases even further, vector does not migrate;

(b) for $h_c < -\epsilon$:

i. if **incremental** ($\Delta\theta_c > 0$) and $\gamma_c > 0$:
$\Delta h_c > 0$, h_c may reach $-\epsilon$, $\gamma_c \cdot \Delta\theta_c = \Delta h_c \leq -\epsilon - h_c$, $\Delta\theta_c \leq \frac{-\epsilon - h_c}{\gamma_c}$ $(E \to S)$

ii. if **incremental** ($\Delta\theta_c > 0$) and $\gamma_c < 0$:
$\Delta h_c < 0$, h_c decreases even further, vector does not migrate;

iii. if **decremental** ($\Delta\theta_c < 0$) and $\gamma_c > 0$:
$\Delta h_c < 0$, h_c decreases even further, vector does not migrate;

iv. if **decremental** ($\Delta\theta_c < 0$) and $\gamma_c < 0$:
$\Delta h_c > 0$, h_c may reach $-\epsilon$, $\gamma_c \cdot \Delta\theta_c = \Delta h_c \leq -\epsilon - h_c$, $\Delta\theta_c \geq \frac{-\epsilon - h_c}{\gamma_c}$ $(E \to S)$

Appendix B

Migration of vectors on varying regularization parameter

1. Migration of the support vectors.

   (a) for $\Delta C < 0$, when C is decreasing (decremental):

      i. for support vector with $0 \leq \theta_s \leq C$, $h_s = -\epsilon$, the variation of C is limited by $0 \leq \theta_s + \Delta\theta_s \leq C + \Delta C$:

         A. for $\beta_s \leq 0$:
            - $\beta_s \cdot \Delta C = \Delta\theta_s \geq 0$, $\theta_s + \Delta\theta_s \geq 0$ is always fulfilled;
            - $\theta_s + \beta_s \cdot \Delta C \leq C + \Delta C$; so $(\beta_s - 1)\Delta C \leq C - \theta_s$ leads to $\Delta C \geq \frac{C-\theta_s}{\beta_s - 1}$, $\Delta\theta_s \geq 0$, θ_s can reach C (migration $S \to E$)

         B. for $0 \leq \beta_s \leq 1$, both conditions are active:
            - $0 \leq \theta_s + \beta_s \Delta C$, leading to $\Delta C \geq -\frac{\theta_s}{\beta_s}$, $\Delta\theta_s \leq 0$, θ_s can reach 0 (migration $S \to O$)
            - $\theta_s + \beta_s \Delta C \leq C + \Delta C$, $(\beta_s - 1)\Delta C \leq C - \theta_s$, leading to $\Delta C \geq \frac{C-\theta_s}{\beta_s - 1}$, $\Delta\theta_s = \beta_s \Delta C$, θ_s can reach C (migration $S \to E$)

         C. for $\beta_s \geq 1$:
            - $\theta_s + \beta_s \Delta_C \geq 0$ leads to $\Delta C \geq -\frac{\theta_s}{\beta_s}$, θ_s decreases (migration $S \to O$)
            - $\theta_s + \beta_s \Delta C \leq C + \Delta C$, or $(\beta_s - 1)\Delta C \leq C - \theta_s$, is always fulfilled.

         Summary:
         - if $\beta_s - 1 \leq 0$, we have $S \to E$ migration:

$$\Delta C \geq \frac{C - \theta_s}{\beta_s - 1}$$

         - if $\beta_s \geq 0$, we have $S \to O$ migration:

$$\Delta C \geq -\frac{\theta_s}{\beta_s}$$

      ii. for support vector with $-C \leq \theta_s \leq 0$, $h_s = +\epsilon$, the variation of C is limited by $-C - \Delta C \leq \theta_s + \Delta\theta_s \leq 0$:

         A. for $\beta_s \leq -1$:
            - $-C - \Delta C \leq \theta_s + \beta_s \Delta C$, or $-C - \theta_s \leq (\beta_s + 1)\Delta C$ is always fulfilled;
            - $\theta_s + \beta_s \Delta C \leq 0$ leads to $\Delta C \geq -\frac{\theta_s}{\beta_s}$, $\Delta\theta_s \geq 0$, θ_s can reach 0 (migration $S \to O$)

         B. for $-1 \leq \beta_s \leq 0$, both conditions are active:
            - $-C - \Delta C \leq \theta_s + \beta_s \Delta C$, leading to $\Delta C \geq \frac{-C-\theta_s}{\beta_s + 1}$ (migration $S \to E$)
            - $\theta_s + \beta_s \Delta C \leq 0$ leads to $\Delta C \geq -\frac{\theta_s}{\beta_s}$ (migration $S \to O$)

         C. for $\beta_s \geq 0$:
            - $-C - \Delta C \leq \theta_s + \beta_s \Delta C$, or $-C - \theta_s \leq (\beta_s + 1)\Delta C$, leading to $\Delta C \geq \frac{-C-\theta_s}{\beta_s + 1}$ (migration $S \to E$)
            - $\theta_s + \beta_s \Delta C \leq 0$ is always fulfilled.

Summary:

- if $\beta_s \leq 0$, we have $S \to O$ migration:

$$\Delta C \geq -\frac{\theta_s}{\beta_s}$$

- if $\beta_s + 1 \geq 0$, we have $S \to E$ migration:

$$\Delta C \geq \frac{-C - \theta_s}{\beta_s + 1}$$

(b) for $\Delta C > 0$, when C is increasing (incremental):

i. for support vector with $0 \leq \theta_s \leq C$, $h_s = -\epsilon$, the variation of C is limited by $0 \leq \theta_s + \Delta\theta_s \leq C + \Delta C$:

A. for $\beta_s \leq 0$:

- $\theta_s + \beta_s \Delta C \geq 0$, or $\beta_s \Delta C \geq -\theta_s$ leads to $\Delta C \leq -\frac{\theta_s}{\beta_s}$ (migration $S \to O$)
- $\theta_s + \beta_s \Delta C \leq C + \Delta C$ or $(\beta_s - 1)\Delta C \leq C - \theta_s$ is always fulfilled;

B. for $0 \leq \beta_s \leq 1$:

- $\theta_s + \beta_s \Delta C \geq 0$, or $\beta_s \Delta C \geq -\theta_s$ is always fulfilled;
- $\theta_s + \beta_s \Delta C \leq C + \Delta C$ or $(\beta_s - 1)\Delta C \leq C - \theta_s$ is also always fulfilled;

C. for $\beta_s \geq 1$:

- $\theta_s + \beta_s \Delta C \geq 0$, or $\beta_s \Delta C \geq -\theta_s$ is always fulfilled;
- $\theta_s + \beta_s \Delta C \leq C + \Delta C$ or $(\beta_s - 1)\Delta C \leq C - \theta_s$ leads to $\Delta C \leq \frac{C - \theta_s}{\beta_s - 1}$ (migration $S \to E$)

Summary:

- if $\beta_s \leq 0$, we have $S \to O$ migration:

$$\Delta C \leq -\frac{\theta_s}{\beta_s}$$

- if $\beta_s - 1 \geq 0$, we have $S \to E$ migration:

$$\Delta C \leq \frac{C - \theta_s}{\beta_s - 1}$$

ii. for support vector with $-C \leq \theta_s \leq 0$, $h_s = +\epsilon$, the variation of C is limited by $-C - \Delta C \leq \theta_s + \Delta\theta_s \leq 0$:

A. for $\beta_s \leq -1$:

- $-C - \Delta C \leq \theta_s + \beta_s \Delta C$, or $-C - \theta_s \leq (\beta_s + 1)\Delta C$ leads to $\Delta C \leq \frac{-C - \theta_s}{\beta_s + 1}$ (migration $S \to E$)
- $\theta_s + \beta_s \Delta C \leq 0$ or $\beta_s \Delta C \leq -\theta_s$ is always fulfilled;

B. for $-1 \leq \beta_s \leq 0$:

- $-C - \Delta C \leq \theta_s + \beta_s \Delta C$, is always fulfilled;
- $\theta_s + \beta_s \Delta C \leq 0$ is also always fulfilled;

C. for $\beta_s \geq 0$:

- $-C - \Delta C \leq \theta_s + \beta_s \Delta C$ is always fulfilled;
- $\theta_s + \beta_s \Delta C \leq 0$ or $\beta_s \Delta C \leq -\theta_s$ leads to $\Delta C \leq -\frac{\theta_s}{\beta_s}$ (migration $S \to O$)

Summary:

- if $\beta_s + 1 \leq 0$, we have $S \to E$ migration:

$$\Delta C \leq \frac{-C - \theta_s}{\beta_s + 1}$$

- if $\beta_s \geq 0$, we have $S \to O$ migration:

$$\Delta C \leq -\frac{\theta_s}{\beta_s}$$

For migration of support vectors, these conditions are summarized in section 4. Hence, we have (conditions may be cumulative):

- for **incremental** ($\Delta C > 0$):
  - if $sgn(\beta_s + sgn(h_s)) \neq sgn(h_s)$, then the limit is imposed by (migration $S \to E$):

$$\Delta C \leq \frac{-sgn(h_s) \cdot C - \theta_s}{\beta_s + sgn(h_s)}$$

  - if $sgn(\beta_s) = sgn(h_s)$, then the limit is imposed by (migration $S \to O$):

$$\Delta C \leq -\frac{\theta_s}{\beta_s}$$

- for **decremental** ($\Delta C < 0$):
  - if $sgn(\beta_s + sgn(h_s)) = sgn(h_s)$, then the limit is imposed by (migration $S \to E$):

$$\Delta C \geq \frac{-sgn(h_s) \cdot C - \theta_s}{\beta_s + sgn(h_s)}$$

  - if $sgn(\beta_s) \neq sgn(h_s)$, then the limit is imposed by (migration $S \to O$):

$$\Delta C \geq -\frac{\theta_s}{\beta_s}$$

2. Migration of other vectors.

   For other (O) vectors, $-\epsilon < h_r < \epsilon$, where $\theta_r = 0$; $\Delta h_r = \gamma_r \cdot \Delta C$.

   (a) for $\Delta C < 0$, when C is decreasing (decremental):
   - i. for $\gamma_r > 0$: $\Delta\gamma_r < 0$, h_r may reach $-\epsilon$, $\gamma_r \Delta C \geq -\epsilon - h_r$, it leads to $\Delta C \geq \frac{-\epsilon - h_r}{\gamma_r}$ (migration $O \to S$)
   - ii. for $\gamma_r < 0$: $\Delta\gamma_r > 0$, h_r may reach ϵ, $\gamma_r \Delta C \leq \epsilon - h_r$, it leads to $\Delta C \geq \frac{\epsilon - h_r}{\gamma_r}$ (migration $O \to S$)

   (b) for $\Delta C > 0$, when C is decreasing (decremental):
   - i. for $\gamma_r > 0$: $\Delta\gamma_r > 0$, h_r may reach ϵ, $\gamma_r \Delta C \geq \epsilon - h_r$, it leads to $\Delta C \leq \frac{\epsilon - h_r}{\gamma_r}$ (migration $O \to S$)
   - ii. for $\gamma_r < 0$: $\Delta\gamma_r < 0$, h_r may reach $-\epsilon$, $\gamma_r \Delta C \geq -\epsilon - h_r$, it leads to $\Delta C \leq \frac{-\epsilon - h_r}{\gamma_r}$ (migration $O \to S$)

   Resuming the $O \to S$ migration:

- for **incremental** ($\Delta C > 0$):

$$\Delta C \leq \frac{sgn(\gamma_r)\epsilon - h_r}{\gamma_r}$$

- for **decremental** ($\Delta C < 0$):

$$\Delta C \geq \frac{-sgn(\gamma_r)\epsilon - h_r}{\gamma_r}$$

3. Migration of error (E) vectors.

   (a) for error vectors with $h_r > \epsilon$, where $\theta_r = -C$; $\Delta h_r = \gamma_r \cdot \Delta C$:

      i. for $\Delta C < 0$ and $\gamma_r < 0$, or $\Delta C > 0$ and $\gamma_r > 0$, then $\Delta h_r > 0$, h_r increases even further, there is no migration;

      ii. for $\Delta C < 0$ and $\gamma_r > 0$, then $\Delta h_r < 0$, h_r may reach ϵ; $\gamma_r \Delta C \geq \epsilon - h_r$ leads to $\Delta C \geq \frac{\epsilon - h_r}{\gamma_r}$ (migration $E \to S$)

      iii. for $\Delta C > 0$ and $\gamma_r < 0$, then $\Delta h_r < 0$, h_r may reach ϵ; $\gamma_r \Delta C \geq \epsilon - h_r$ leads to $\Delta C \leq \frac{\epsilon - h_r}{\gamma_r}$ (migration $E \to S$)

   (b) for error vectors with $h_r < -\epsilon$, where $\theta_r = C$; $\Delta h_r = \gamma_r \cdot \Delta C$:

      i. for $\Delta C < 0$ and $\gamma_r > 0$, or $\Delta C > 0$ and $\gamma_r < 0$, then $\Delta h_r < 0$, h_r decreases even further, there is no migration;

      ii. for $\Delta C < 0$ and $\gamma_r < 0$, then $\Delta h_r > 0$, h_r may reach $-\epsilon$; $\gamma_r \Delta C \leq -\epsilon - h_r$ leads to $\Delta C \geq \frac{-\epsilon - h_r}{\gamma_r}$ (migration $E \to S$)

      iii. for $\Delta C > 0$ and $\gamma_r > 0$, then $\Delta h_r > 0$, h_r may reach $-\epsilon$; $\gamma_r \Delta C \leq -\epsilon - h_r$ leads to $\Delta C \leq \frac{-\epsilon - h_r}{\gamma_r}$ (migration $E \to S$)

   Resuming the $E \to S$ migration:

   - for **incremental** ($\Delta C > 0$):
     (a) if $sgn(h_r) \neq sgn(\gamma_r)$:

$$\Delta C \leq \frac{-sgn(\gamma_r)\epsilon - h_r}{\gamma_r}$$

     (b) if $sgn(h_r) = sgn(\gamma_r)$, the limit is not active;
   - for **decremental** ($\Delta C < 0$):
     (a) if $sgn(h_r) = sgn(\gamma_r)$:

$$\Delta C \geq \frac{sgn(\gamma_r)\epsilon - h_r}{\gamma_r}$$

     (b) if $sgn(h_r) \neq sgn(\gamma_r)$, the limit is not active;

4. Variation of C is also limited by a target value C_{target}:

   - for **incremental**:

$$\Delta C \leq C_{target} - C$$

   - for **decremental**:

$$\Delta C \geq C_{target} - C$$