On the Convergence of *SVMTorch*, an Algorithm for Large-Scale Regression Problems

Ronan Collobert [1] Samy Bengio [2]

IDIAP–RR 00-24

August 17, 2000

[1] IDIAP, CP 592, 1920 Martigny, Switzerland, collober@idiap.ch

[2] IDIAP, CP 592, 1920 Martigny, Switzerland, bengio@idiap.ch

IDIAP Research Report 00-24

On the Convergence of *SVMTorch*, an Algorithm for Large-Scale Regression Problems

Ronan Collobert Samy Bengio

August 17, 2000

Abstract. Recently, many researchers have proposed decomposition algorithms for SVM regression problems (see for instance [11, 3, 6, 10]). In a previous paper [1], we also proposed such an algorithm, named *SVMTorch*. In this paper, we show that while there is actually no convergence proof for any other decomposition algorithm for SVM regression problems to our knowledge, such a proof does exist for *SVMTorch* for the particular case where no shrinking is used and the size of the working set is equal to 2, which is the size that gave the fastest results on most experiments we have done. This convergence proof is in fact mainly based on the convergence proof given by Keerthi and Gilbert [4] for their SVM classification algorithm.

1 Introduction

Vapnik has proposed in [12] a method to solve regression problems using Support Vector Machines (SVMs). It has yielded excellent performances on many regression and time series prediction problems (see for instance [7, 2]). Recently, we have proposed a fast decomposition algorithm for large-scale regression problems [1] using SVMs. Unlike other decomposition algorithms for regression problems (see for instance [11, 3, 6, 10]), there exists a convergence proof for our algorithm, and the goal of this paper is to show such a proof. Let us first recall the general problem of SVM for regression and our method to solve it.

Given a training set of l *examples* $(\boldsymbol{x}_i, y_i)$ with $\boldsymbol{x}_i \in E$ and $y_i \in \mathbb{R}$, where E is an Euclidean space with a scalar product denoted $(\ \cdot\ )$, we want to estimate the following linear regression:

$$f(\boldsymbol{x}) = (\boldsymbol{w} \cdot \boldsymbol{x}) + b$$

(with $b \in \mathbb{R}$) with a precision ϵ. For this, we minimize

$$\frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l} |y_i - f(\boldsymbol{x}_i)|_\epsilon$$

where $\frac{1}{2}\|\boldsymbol{w}\|^2$ is a regularization factor, C is a fixed constant, and $|.|_\epsilon$ is the ϵ-insensitive loss function defined by Vapnik:

$$|z|_\epsilon = \max\{0, |z| - \epsilon\}.$$

Written as a constrained optimization problem, it amounts to minimizing

$$\tau(\boldsymbol{w}, \boldsymbol{\xi}, \boldsymbol{\xi}^\star) = \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{l}(\xi_i + \xi_i^\star)$$

subject to

$$((\boldsymbol{w} \cdot \boldsymbol{x}_i) + b) - y_i \leq \epsilon + \xi_i \tag{1}$$

$$y_i - ((\boldsymbol{w} \cdot \boldsymbol{x}_i) + b) \leq \epsilon + \xi_i^\star \tag{2}$$

$$\xi_i,\ \xi_i^\star \geq 0.$$

To generalize to non-linear regression, we replace the dot product with a kernel [1] $k(\cdot)$. Then, introducing Lagrange multipliers $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}^\star$, the optimization problem can be stated as:

Minimize the function

$$\mathcal{W}(\boldsymbol{\alpha}, \boldsymbol{\alpha}^\star) = \frac{1}{2}(\boldsymbol{\alpha}^\star - \boldsymbol{\alpha})^{\mathrm{T}} K(\boldsymbol{\alpha}^\star - \boldsymbol{\alpha}) - (\boldsymbol{\alpha}^\star - \boldsymbol{\alpha})^{\mathrm{T}}\boldsymbol{y} + \epsilon(\boldsymbol{\alpha}^\star + \boldsymbol{\alpha})^{\mathrm{T}}\mathbf{1} \tag{3}$$

subject to

$$(\boldsymbol{\alpha} - \boldsymbol{\alpha}^\star)^{\mathrm{T}}\mathbf{1} = 0 \tag{4}$$

and

$$0 \leq \alpha_i^\star,\ \alpha_i \leq C, \quad i = 1...l \tag{5}$$

where K is the matrix with coefficients $K_{ij} = k(\boldsymbol{x}_i, \boldsymbol{x}_j)$. The estimate of the regression function at a given point is then

$$f(\boldsymbol{x}) = \sum_{i=1}^{l}(\alpha_i^\star - \alpha_i)k(\boldsymbol{x}_i, \boldsymbol{x}) + b$$

[1] note that this kernel needs to verify the Mercer's conditions in order for convergence proofs to work.

where b is computed using the fact that (1) becomes an equality with $\xi_i = 0$ if $0 < \alpha_i < C$ and (2) becomes an equality with $\xi_i^\star = 0$ if $0 < \alpha_i^\star < C$.

Let us denote

$$\boldsymbol{\beta} = \begin{pmatrix} \boldsymbol{\alpha} \\ -\boldsymbol{\alpha}^\star \end{pmatrix}$$

and

$$\tilde{K} = \begin{pmatrix} K & K \\ K & K \end{pmatrix}$$

as well as

$$\boldsymbol{b} = \begin{pmatrix} -\boldsymbol{y} - \mathbf{1}\,\epsilon \\ -\boldsymbol{y} + \mathbf{1}\,\epsilon \end{pmatrix}.$$

The optimization problem is thus (see [12, 1] for more details) :

$$\tilde{\mathcal{W}}(\boldsymbol{\beta}) = \frac{1}{2}\boldsymbol{\beta}^{\mathrm{T}}\tilde{K}\boldsymbol{\beta} - \boldsymbol{\beta}^{\mathrm{T}}\boldsymbol{b} \tag{6}$$

subject to

$$\boldsymbol{\beta}^{\mathrm{T}}\,\mathbf{1} = 0 \tag{7}$$

and

$$0 \le \delta_i\,\beta_i \le C, \quad i = 1...2l \tag{8}$$

where $\delta_i = 1$ for $1 \le i \le l$ and $\delta_i = -1$ for $l + 1 \le i \le 2l$.

Solving this optimization problem with a decomposition method (as proposed for instance by Osuna *et al* [8]) consists in an iterative procedure where at each iteration, one selects a set of variables in $\{\beta_1, \ldots, \beta_{2l}\}$ and then minimizes (6) with respect to the selected variables. Note that this selection scheme is particular to *SVMTorch*, while all the other decomposition algorithms we have seen for regression problems select pairs of variables $(\alpha_i, \alpha_i^\star)$. This apparently small difference enables us to show a convergence proof for our algorithm.

In the particular case where the number of selected variables at each iteration is fixed to 2, one can then use a analytical method to solve the subproblem, as we have shown in [1] and was also proposed for the *SMO* algorithm [9].

Going back to our method proposed in [1], in the case where no *shrinking*[2] is done and with the number of selected variables set to 2, our algorithm can then be written as:

Algorithm 1 *Given a* $\tau > 0$,

1. *Set* $\boldsymbol{\beta} = \mathbf{0}$.

2. *Select variables* β_{i_0} *and* β_{i_1} *where* i_0 *and* i_1 *verify*

$$\begin{array}{l} \delta_{i_0}\mathcal{W}'_{i_0}(\boldsymbol{\alpha}, \boldsymbol{\alpha}^\star) = \max_{i \in A_1} \delta_i \mathcal{W}'_i(\boldsymbol{\alpha}, \boldsymbol{\alpha}^\star) \\ \delta_{i_1}\mathcal{W}'_{i_1}(\boldsymbol{\alpha}, \boldsymbol{\alpha}^\star) = \min_{i \in A_2} \delta_i \mathcal{W}'_i(\boldsymbol{\alpha}, \boldsymbol{\alpha}^\star) \end{array} \tag{9}$$

where again $\delta_i = 1$ *for* $1 \le i \le l$ *and* $\delta_i = -1$ *for* $l + 1 \le i \le 2l$, *and*

$$A_1 = \{i \ : \ 0 < \beta_i \le C \textit{ for } i \le l\} \cup \{i \ : \ 0 \le \delta_i\beta_i < C \textit{ for } i > l\}$$

$$A_2 = \{i \ : \ 0 \le \beta_i < C \textit{ for } i \le l\} \cup \{i \ : \ 0 < \delta_i\beta_i \le C \textit{ for } i > l\}.$$

3. *Solve analytically the optimization problem (6) with respect to these two variables. Update* $\boldsymbol{\beta}$ *with respect to the obtained solution.*

[2] *Shrinking* is a heuristic used to eliminate variables that tend to be stuck at bounds 0 or C for many iterations.

4. Verify the optimality conditions

$$\begin{array}{rc} \textit{for } i \textit{ such that } 0 < \delta_i\,\beta_i < C: & \hat{\lambda}^{eq} - \tau \leq -\delta_i \mathcal{W}_i^{'}(\boldsymbol{\alpha},\,\boldsymbol{\alpha}^{\star}) \leq \hat{\lambda}^{eq} + \tau \\ \textit{for } i \textit{ such that } \beta_i = 0: & \mathcal{W}_i^{'}(\boldsymbol{\alpha},\,\boldsymbol{\alpha}^{\star}) + \delta_i \hat{\lambda}^{eq} \geq -\tau \\ \textit{for } i \textit{ such that } \delta_i\beta_i = C: & \mathcal{W}_i^{'}(\boldsymbol{\alpha},\,\boldsymbol{\alpha}^{\star}) + \delta_i \hat{\lambda}^{eq} \leq \tau \end{array} \qquad (10)$$

where

$$\hat{\lambda}^{eq} = \frac{1}{|A \cup B|}\left(\sum_{i \in B} \mathcal{W}_{i+l}^{'}(\boldsymbol{\alpha},\,\boldsymbol{\alpha}^{\star}) - \sum_{i \in A} \mathcal{W}_i^{'}(\boldsymbol{\alpha},\,\boldsymbol{\alpha}^{\star})\right) \qquad (11)$$

with

$$A = \{i,\ 0 < \alpha_i < C\}, \quad B = \{i,\ 0 < \alpha_i^{\star} < C\}.$$

When all optimality conditions are verified, stop the algorithm; otherwise, go back to step 2.

The aim of this paper is to show that such an algorithm converges. In section 2, we expose a recent convergence theorem from Keerthi and Gilbert [4] while in section 3 we show how it applies to our algorithm.

2 The Convergence Theorem of Keerthi and Gilbert

In a recent paper, Keerthi and Gilbert [4] showed a convergence theorem for a modified version of *SMO* given by Keerthi *et al* [5] for SVM classification problems. In fact, in their paper, Keerthi and Gilbert talk about the general case where one wants to minimize

$$f(\boldsymbol{\eta}) = \frac{1}{2}\boldsymbol{\eta}^{\mathrm{T}} Q \boldsymbol{\eta} + \boldsymbol{p}^{\mathrm{T}} \boldsymbol{\eta} \qquad (12)$$

subject to

$$a_i \leq \eta_i \leq b_i \ \forall i \ \text{ and } \sum_i z_i \eta_i = c$$

where Q is symmetric positive semi-definite, $a_i < b_i \ \forall i$ and $z_i \neq 0 \ \forall i$. They call $\mathcal{F}$ the feasible set of the quadratic problem. They suppose $\mathcal{F}$ non-empty, and f bounded below on $\mathcal{F}$. They denote

$$F_i(\boldsymbol{\eta}) = ([Q\boldsymbol{\eta}]_i + p_i)/z_i$$

and define the sets

$$\begin{array}{rcl} I_0(\boldsymbol{\eta}) & = & \{i : a_i < \eta_i < b_i\} \\ I_1(\boldsymbol{\eta}) & = & \{i : z_i > 0,\ \eta_i = a_i\} \\ I_2(\boldsymbol{\eta}) & = & \{i : z_i < 0,\ \eta_i = b_i\} \\ I_3(\boldsymbol{\eta}) & = & \{i : z_i > 0,\ \eta_i = b_i\} \\ I_4(\boldsymbol{\eta}) & = & \{i : z_i < 0,\ \eta_i = a_i\} \\ & & \\ I_{up}(\boldsymbol{\eta}) & = & I_0(\boldsymbol{\eta}) \cup I_1(\boldsymbol{\eta}) \cup I_2(\boldsymbol{\eta}) \\ I_{low}(\boldsymbol{\eta}) & = & I_0(\boldsymbol{\eta}) \cup I_3(\boldsymbol{\eta}) \cup I_4(\boldsymbol{\eta}) \end{array}$$

Moreover, they define $(i,\,j)$ as being a *violating pair* if one of the following conditions is verified:

$$\begin{array}{llll} i \in I_{up}(\boldsymbol{\eta}), & j \in I_{low}(\boldsymbol{\eta}) & \textit{and} & F_i(\boldsymbol{\eta}) < F_j(\boldsymbol{\eta}) - \tau \\ i \in I_{low}(\boldsymbol{\eta}), & j \in I_{up}(\boldsymbol{\eta}) & \textit{and} & F_i(\boldsymbol{\eta}) > F_j(\boldsymbol{\eta}) + \tau. \end{array}$$

They then prove that *the following algorithm stops after a finite number of iterations* :

Algorithm 2 (GSMO) *Given a* $\tau > 0$

1. *Choose some* $\boldsymbol{\eta} \in \mathcal{F}$.
2. *If* $\boldsymbol{\eta}$ *satisfies*

$$\min_{i \in I_{up}(\boldsymbol{\eta})} F_i(\boldsymbol{\eta}) \geq \max_{i \in I_{low}(\boldsymbol{\eta})} F_i(\boldsymbol{\eta}) - \tau \tag{13}$$

   then stop.

3. *Choose* (j_0, j_1) *a* τ*-violating pair. Minimize* f *on* $\mathcal{F}$ *while varying only* η_{j_0} *and* η_{j_1}. *Set* $\boldsymbol{\eta}$ *to the point thus obtained. Go to step* 2.

3 Convergence of our Algorithm

Using the previous notation, let $\boldsymbol{\eta} = \boldsymbol{\beta}$, $\boldsymbol{p} = -\boldsymbol{b}$, $Q = \tilde{K}$, $z_i = 1\ \forall i$, $c = 0$, and

$$\begin{array}{lll} a_i = 0, & b_i = C & \text{for } 0 \leq i \leq l \\ a_i = -C, & b_i = 0 & \text{for } (l+1) \leq i \leq 2l. \end{array}$$

Our optimization problem (6) is then totally equivalent to the one from Keerthi and Gilbert (12). Moreover, it is easy to see that the hypothesis needed by Keerthi and Gilbert are verified: the initial solution $\boldsymbol{\beta} = \mathbf{0}$ is in $\mathcal{F}$, K is positive semi-definite for a kernel k verifying Mercer's conditions, and *a fortiori* $\tilde{K}$ is also positive semi-definite. Moreover $z_i \neq 0\ \forall i$, $\mathcal{F}$ is clearly non-empty and compact. f is thus bounded below.

Let us now show that our stopping conditions (10) are *weaker* than those from Keerthi and Gilbert (13). It is easy to see that

$$F_i(\boldsymbol{\eta}) = \delta_i \mathcal{W}_i^{'}(\boldsymbol{\alpha}, \boldsymbol{\alpha}^{\star})$$

and thus one can easily deduce that the conditions (10) are equivalent to

$$\begin{array}{ll} \forall i \in I_0(\boldsymbol{\eta}): & \hat{\lambda}^{eq} - \tau \leq -F_i(\boldsymbol{\eta}) \leq \hat{\lambda}^{eq} + \tau \\ \forall i \in I_1(\boldsymbol{\eta}): & F_i(\boldsymbol{\eta}) \geq -\hat{\lambda}^{eq} - \tau \\ \forall i \in I_3(\boldsymbol{\eta}): & F_i(\boldsymbol{\eta}) \leq -\hat{\lambda}^{eq} + \tau. \end{array} \tag{14}$$

Hence if $\boldsymbol{\eta}$ verifies the stopping conditions (13), and taking into account the fact that

$$-\max_{i \in I_{low}(\boldsymbol{\eta})} F_i(\boldsymbol{\eta}) \leq \hat{\lambda}^{eq} \leq -\min_{i \in I_{up}(\boldsymbol{\eta})} F_i(\boldsymbol{\eta})$$

then

$$\begin{array}{lll} \forall i \in I_{low}(\boldsymbol{\eta}) \quad F_i(\boldsymbol{\eta}) & \leq & \min_{i \in I_{up}(\boldsymbol{\eta})} F_i(\boldsymbol{\eta}) + \tau \\ & \leq & -\hat{\lambda}^{eq} + \tau \end{array}$$

and

$$\begin{array}{lll} \forall i \in I_{up}(\boldsymbol{\eta}) \quad -F_i(\boldsymbol{\eta}) & \leq & -\max_{i \in I_{low}(\boldsymbol{\eta})} F_i(\boldsymbol{\eta}) + \tau \\ & \leq & \hat{\lambda}^{eq} + \tau \end{array}$$

which implies the conditions (14). In other words, our stopping conditions are weaker than those from Keerthi and Gilbert.

Finally, noting that in our selected variables (9) one has[3]

$$A_1 = I_0(\boldsymbol{\eta}) \cup I_3(\boldsymbol{\eta}) = I_{low}(\boldsymbol{\eta})$$

$$A_2 = I_0(\boldsymbol{\eta}) \cup I_1(\boldsymbol{\eta}) = I_{up}(\boldsymbol{\eta})$$

[3] Indeed, one can see that in our case, one has always $z_i > 0$ and thus $I_2(\boldsymbol{\eta})$ and $I_4(\boldsymbol{\eta})$ are empty sets.

one can see that we selected β_{i_0} and β_{i_1} such that

$$F_{i_0}(\boldsymbol{\eta}) = \max_{i \in I_{low}(\boldsymbol{\eta})} F_i(\boldsymbol{\eta})$$

$$F_{i_1}(\boldsymbol{\eta}) = \min_{i \in I_{up}(\boldsymbol{\eta})} F_i(\boldsymbol{\eta}).$$

Moreover, if our algorithm has not stopped yet, in other words if our stopping conditions have not been verified yet, then since our stopping conditions are weaker than those from Keerthi and Gilbert, the latter are not verified either. Hence one then have

$$\min_{i \in I_{up}(\boldsymbol{\eta})} F_i(\boldsymbol{\eta}) < \max_{i \in I_{low}(\boldsymbol{\eta})} F_i(\boldsymbol{\eta}) - \tau$$

and thus

$$F_{i_1}(\boldsymbol{\eta}) < F_{i_0}(\boldsymbol{\eta}) - \tau$$

and (i_0, i_1) is a *violating pair* (in fact it is the *worst violating pair*).

Our algorithm is thus a special case of the general algorithm proposed by Keerthi and Gilbert. It verifies all their hypothesis, excepted that our stopping criterion is weaker. Thus, our algorithm converges.

4 Conclusion

In this paper, we have shown that *SVMTorch*, a decomposition algorithm proposed to solve large-scale regression problems using support vectors machines [1], converges when the size of the subproblem is set to 2 and no *shrinking* is done. We showed this convergence using a general theorem recently given by Keerthi and Gilbert [4]. Finally, even if there is no proven convergence when using *shrinking*, empirical experiments [1] showed that it does speed up a lot convergence times.

References

[1] R. Collobert and S. Bengio. Support vector machines for large-scale regression problems. IDIAP-RR 17, IDIAP, 2000. Available at `ftp://www.idiap.ch/pub/reports/2000/rr00-17.ps.gz`.

[2] H. Drucker, C.J.C. Burges, L. Kaufman, A. Smola, and V. Vapnik. Support vector regression machines. In M. Mozer, M. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, pages 155–161. MIT Press, 1997.

[3] G.W. Flake and S. Lawrence. Efficient SVM regression training with SMO. Submitted to Machine Learning. Available at `http://external.nj.nec.com/homepages/flake/smorch.ps`.

[4] S.S. Keerthi and E.G. Gilbert. Convergence of a generalized SMO algorithm for SVM classifier design. Technical Report CD-00-01, Control Division, Dept. of Mechanical and Production Engineering, National University of Singapore, 2000. Available at `http://guppy.mpe.nus.edu.sg/~mpessk/svm/conv_ml.ps.gz`.

[5] S.S. Keerthi, S.K. Shevade, C. Bhattacharyya, and K.R.K. Murthy. Improvements to platt's SMO algorithm for SVM classifier design. Technical Report CD-99-14, Control Division, Dept. of Mechanical and Production Engineering, National University of Singapore, 1999. To appear in Neural Computation. Available at `http://guppy.mpe.nus.edu.sg/~mpessk/smo_mod.ps.gz`.

[6] P. Laskov. An improved decomposition algorithm for regression support vector machines. In S.A. Solla, T.K. Leen, and K.-R. Müller, editors, *Advances in Neural Information Processing Systems 12*. MIT Press, 2000. Available at `http://www.cis.udel.edu/~laskov/publications/NIPS-99.ps.gz`.

[7] K.-R. Müller, A. Smola, G. Rätsch, B. Schölkopf, J. Kohlmorgen, and V. Vapnik. Predicting time series with support vector machines. In W. Gerstner, A. Germond, M. Hasler, and J.-D. Nicoud, editors, *Artificial Neural Networks - ICANN'97*, pages 999–1004. Springer, 1997.

[8] Edgar Osuna, Robert Freund, and Federico Girosi. An improved training algorithm for support vector machines. In J. Principe, L. Giles, N. Morgan, and E. Wilson, editors, *Neural Networks for Signal Processing VII - Proceedings of the 1997 IEEE Workshop*, pages 276–285. IEEE, New York, 1997.

[9] John C. Platt. Fast training of support vector machines using sequential minimal optimization. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods*. MIT Press, 1999.

[10] S.K. Shevade, S.S. Keerthi, C. Bhattacharyya, and K.R.K. Murthy. Improvements to SMO algorithm for SVM regression. Technical Report CD-99-16, Control Division, Dept. of Mechanical and Production Engineering, National University of Singapore, 1999. To appear in IEEE Transaction on Neural Networks. Available at `http://guppy.mpe.nus.edu.sg/~mpessk/smoreg_mod.shtml`.

[11] A. J. Smola and B. Schölkopf. A tutorial on support vector regression. Technical Report NeuroCOLT Technical Report NC-TR-98-030, Royal Holloway College,University of London, UK, 1998.

[12] Vladimir N. Vapnik. *The nature of statistical learning theory.* Springer, second edition, 1995.