Proceedings of the V Brazilian Conference on Neural Networks - V Congresso Brasileiro de Redes Neurais
pp. 661–666, April 2–5, 2001 - , Rio de Janeiro - RJ - Brazil

APLS: A Fast Approximate Algorithm for Partial Least-Squares

Ruy Luiz Milidiú
Raúl Rentería
Departamento de Informática
Pontifícia Universidade Católica PUC-Rio
Rio de Janeiro - Brasil
milidiu@inf.puc-rio.br, renteria@inf.puc-rio.br

Abstract

APLS, an approximate algorithm for solving the Partial Least-Squares problem is presented. When dealing with more than one dependent variable in the standard PLS modeling, one must compute eigenvectors of the mixed independent and dependent variables matrix, thus introducing a time overhead in the algorithm. APLS provides an approximate method for this calculation with competitive modeling error and improved running time. In our experiments, APLS shows a 30% faster running time when compared to the standard PLS algorithm.

1. Introduction

The PLS algorithm [1] has been widely used as a chemometric tool for Near-Infrared spectral analysis [2, 3, 4]. The simplicity of the technique and robustness of the generated model, also make the partial least-squares approach a powerful tool for factor analysis, being applied to many other areas such as process monitoring, marketing analysis and image processing [5, 6, 7].

In this paper, we propose APLS, an approximate algorithm for solving the Partial Least-Squares problem. In the classical PLS modeling, the eigenvectors of the mixed independent and dependent variables matrix must be computed in the case of two or more dependent variables. This can be accomplished for example using neural techniques such as Hebbian learning [8]. APLS provides an approximate method for this calculation, not relying on any convergence criteria.

In order to measure the performance of APLS, we report some experiments with the wheat data set for chemometrics [9] and two other data sets containing combustible NIR spectra [10]. The APLS model shows a competitive quality when compared to the standard PLS model. Similar prediction errors and corresponding number of factors were observed. Moreover, since it is not a convergence dependent technique, it shows a 30 % faster running time.

In section 2, we present the standard PLS algorithm. In section 3, our approximate version is described. In section 4 a running time analysis of the algorithm is made, and in section 5 the empirical results obtained with the selected data set are shown.

2 PLS Algorithm

2.1 Modeling

Partial Least Squares (PLS) is a multivariate statistical method, based on the use of factors, which is aimed at prediction [1]. The goal is to predict the values of a set of variables y based on the observed values of a set of variables x. As an example, x may be formed by the values of a time series window and y be taken as the value of a single future observation, or as the values of a set of future points in the same time series.

The construction of a PLS model requires a set of observation samples (patterns) and also their respective future values. Let X be the matrix containing in its rows the patterns of observations and Y be the matrix containing in its rows the effective values to be predicted.

The PLS method is a modeling procedure that simultaneously estimates underlying factors in both X and Y. These factors are then used to define a subspace in X that is more adequate to model Y. With PCR [4], the rotation defined by the eigenvectors is used to find a subspace in X that subsequently is used to model Y. The approach taken by PLS is very similar to that of Principal Component Analysis (PCA) [11], except that factors are chosen to describe the variables in Y as well as in X. This is accomplished by using the columns of the Y matrix to estimate the factors for X. At the same time, the columns of X are used to estimate the factors of Y. The resulting models are

$$X = TP + E$$

$$Y = UQ + F$$

where the elements of T and U are called the scores of X and Y, respectively, and the elements of P and Q are called the loadings. The matrices E and F are the errors associated with modeling X and Y with the PLS model.

The T factors are not optimal for estimating the columns of X as is the case with PCA , but are rotated so as to simultaneously describe the Y matrix.

In the ideal situation, the sources of variation in X are exactly equal to the sources of variation in Y, and the factors for X and Y are identical. In real applications, X varies in ways not correlated to the variation in Y, and therefore $t \neq u$ (t and u are columns of T and U, respectively). However, when both matrices are used to estimate factors, the factors for the X and Y matrices have the following relationship:

$$u = bt + \epsilon$$

where b is termed the inner relationship between u and t and is used to calculate subsequent factors if the intrinsic dimensionality of X is greater than one. Geometrically, this states that the vectors u and t are approximately equal except for their lengths.

The main advantage of PLS is that it incorporates more information in the model-building phase.

2.2 The Algorithm

The Partial Least-Squares algorithm, as described in [1], can be decomposed in the following steps:

1. given a data set for training, a regression model is built. This is the calibration or training step;

2. given an independent data set, called test set, predictions are made using the model that has just been built.

In the following subsections these two steps are explained.

2.2.1 Model Training

The standard PLS algorithm, as described in figure 1, uses as the training set the $n \times m$ matrix X, and the $n \times l$ matrix Y. Observe that X contains the n observations of m independent variables. Y contains the corresponding values for the dependent variables.

At each iteration, the following factors are calculated regarding the regression model between the scores for X and Y:

1. the weights w_i;

2. the regression vector coefficients b_i for the inner relation;

3. the loadings represented by p_i.

As already stated, there is a key difference between PLS and other regression methods [1] such as PCR. Both methods construct a regression on principal components, however the model constructed by PLS also uses information from the dependent variable Y to bias the principal components. In fact, as we can see in the second and third lines of figure 1, the weighting factor w_i being an eigenvector of $X^\top YY^\top X$, provides a better quality for the prediction step.

```
Standard PLS
for  $i = 1$ to $m$
     $R \leftarrow$ eigenvector of $X^\top YY^\top X$
         associated with the
         largest eigenvalue
     $w_i \leftarrow R/||R||$

     $t_i \leftarrow Xw_i$
     $b_i^\top \leftarrow t_i^\top Y/t_i^\top t_i$

     $p_i \leftarrow X^\top t_i$
     $p_i \leftarrow p_i/||p_i||$

     $X \leftarrow X - t_i p_i^\top$
     $Y \leftarrow Y - t_i b_i^\top$
end
```

Figure 1: Standard PLS algorithm.

```
PLS Prediction
for  $i = 1$ to $k$
     $t_i \leftarrow Xw_i$
     $y \leftarrow y + t_i b_i^\top$
     $X \leftarrow X - t_i p_i^\top$
end
```

Figure 2: Standard PLS algorithm for the prediction of y.

2.2.2 Prediction Step

Given a trained model, obtained as described in the previous section, one can make predictions by using an independent data set X. Figure 2 shows the algorithm for this step. It should be noticed that the number of desired factors for the prediction is indicated by the variable k.

A common procedure, when determining the optimal number of factors k to be used in the prediction, consists in calculating a statistic for the lack of model accuracy called PRESS [1] (Prediction Residual Sum of Squares) which is sum of the squared prediction error using k factors. This kind of method [4] uses an independent data set X with an already known variable Y. PRESS is calculated for each value of k, and the one that yields the minimum PRESS indicates the recommended number of factors to be chosen.

3. APLS: Approximate PLS Algorithm

As shown in the previous section, the eigenvector of the matrix $X^\top YY^\top X$ associated with the largest eigenvalue must be computed for each desired factor. One could use neural network techniques such as Hebbian learning [8], based on a learning rate and thus providing an iterative algorithm with a convergence criterion. APLS substitute this step by an approximate non-iterating one.

3.1 The Algorithm

Let G be the matrix $X^\top YY^\top X$. For each factor, we must find the eigenvector w of G associated with the largest eigenvalue.

When Y has only one dependent variable ($l = 1$) the matrix G has rank 1 and the corresponding eigenvector is simply $X^\top Y$. This eigenvector corresponds to the unique non-zero eigenvalue $(X^\top Y)^\top (X^\top Y)$. On the other hand, when $l \geq 2$ that is not true anymore. Nevertheless, one can decompose the matrix G in the following way:

$$G = X^\top(Y_1Y_1^\top + Y_2Y_2^\top + \cdots + Y_lY_l^\top)X$$

and then,

$$G = X^\top Y_1Y_1^\top X + \cdots + X^\top Y_lY_l^\top X \qquad (1)$$

where Y_i $(1 \leq i \leq d)$ corresponds to the ith column of Y. For a simpler notation, we write (1) as

$$G = G_1 + G_2 + \cdots + G_l$$

where $G_i = X^\top Y_iY_i^\top X$ for $i = 1, \ldots, l$.

As we can see, the eigenvector w of G associated with the largest eigenvalue is also an eigenvector of the sum of the l already known matrices G_i of rank 1. It is acceptable that the eigenvector w_i of matrix G_i with the largest eigenvalue λ_i will have a greater influence on w, and that eigenvectors having little correlation with w_i (towards orthogonality) will have little contribution to the sum described in (1). It follows that a simple approximation for w can be obtained through the following scheme:

1. For each $i = 1, \ldots, l$, calculate the eigenvector w_i and eigenvalue λ_i of G_i given by the following relations
$$w_i = X^\top Y_i$$
$$\lambda_i = w_i^\top w_i$$

2. Find the eigenvector $w_{(1)}$ corresponding to the largest eigenvalue $\lambda_{(1)}$, defined by
$$\lambda_{(1)} = \max_{1\leq i\leq l}\{\lambda_i\}$$

3. Calculate w by
$$w = \sum_{i=1}^{l} \lambda_i w_i (w_i^\top w_{(1)}) \qquad (2)$$
and apply normalization $w \leftarrow w/||w||$.

Equation (2) offers an explicit form for the computation of w, not requiring any kind of convergence criteria such as presented by Hebbian learning or PLS.

The results obtained show an algorithm with a competitive quality of prediction and an improved running time, as will be detailed in the following sections.

4. Running Time Analysis

Let X and Y be $(n \times m)$ and $(n \times l)$ matrices respectively. For both PLS and APLS, considering the code presented in Figure 1, we can describe each factor calculation by the two steps that follows:

1. Compute the eigenvector w associated with the largest eigenvalue. This corresponds to the first two assignments in Figure 1;

2. Compute the regression vector coefficients b, loadings p and residuals matrices. This corresponds to the remaining assignments in Figure 1.

Hence, the time complexity of PLS and APLS differ only in the first step just described.

For PLS, we use the NIPALS algorithm presented in [1], which is a power method for the largest eigenvector extraction [12]. At each step towards convergence, it is easily found that a $O(nml)$ time complexity is obtained.

On the other hand, APLS has the same $O(nml)$ time complexity.

We can conclude that in the case of NIPALS requiring only one iteration, the time complexity of both PLS and APLS are the same. However, this does not occur often. Usually, NIPALS requires more than 10 iterations to converge at usual error levels. This makes the APLS approach faster since it requires no iteration.

In fact, as shown in [12], the error obtained at the kth iteration of NIPALS for the eigenvector estimate $w^{(k)}$ is bound by

$$|\sin(\theta_k)| \leq \tan(\theta_0) \left|\frac{\lambda_{(2)}}{\lambda_{(1)}}\right|^k$$

where

$$cos(\theta_k) = \left|w_{(1)}^\top w^{(k)}\right|$$

and $\lambda_{(1)}$ and $\lambda_{(2)}$ are the largest and second largest eigenvalues. As we can see, depending of the rate $|\lambda_{(2)}|/|\lambda_{(1)}|$ the convergence rate can be very slow, making the APLS approach relatively faster.

5. Experimental results

We have tested the APLS algorithm on three data sets. The first one was taken from Kalivas [9]. We used the data set containing the NIR spectra of 100 wheat samples along with specified protein and moisture content. Samples were measured using diffuse reflectance as $\log(1/R)$ from 1100 to 2500 nm in 2nm intervals. Of the 100 spectra, 70 were utilized for training (calibration) and the 30 remaining for testing (validation) the constructed model. Spectra were reduced to contain only 141 response by using every fifth response and then mean centered.

When comparing the PRESS of the models produced by standard PLS and APLS we obtained the curves shown in Figure 3.

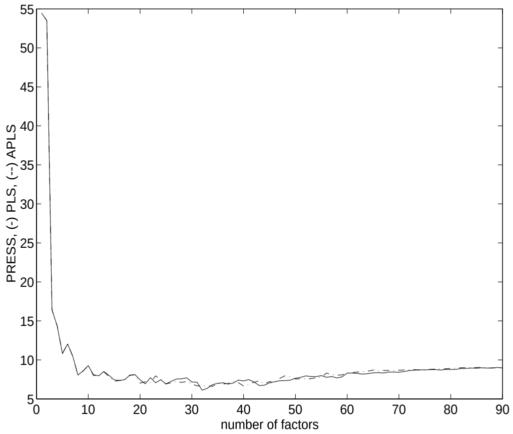

Figure 3: PRESS values for PLS and APLS, data set 1.

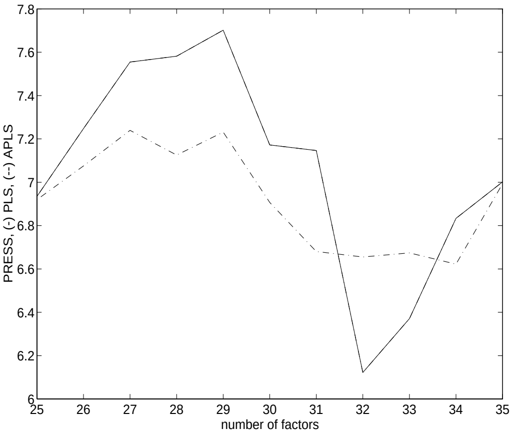

Figure 4: PRESS values of PLS and APLS for data set 1, around the selected factors.

For a better analysis we plotted in Figure 4 the critical region containing the optimal number of factors.

As the second data set, we used the light gas oil data available at Dalhousie University [10]. This set is for the calibration of light gas oil (and diesel) fuels for hydrocarbon content and consists of 115 samples from three subsets for which the spectra over 572 channels have been obtained. For the calibration and validation matrices we used the first 70 and remaining 44 samples respectively, along with the concentrations of the four components in each sample. Being an outlier, the last sample (115) was not used.
In Figure 5 we show the performance of PLS and APLS for the first 100 factors.

Again, for a better understanding, Figure 6 shows a detailed view in the range of factors from 1 to 10.

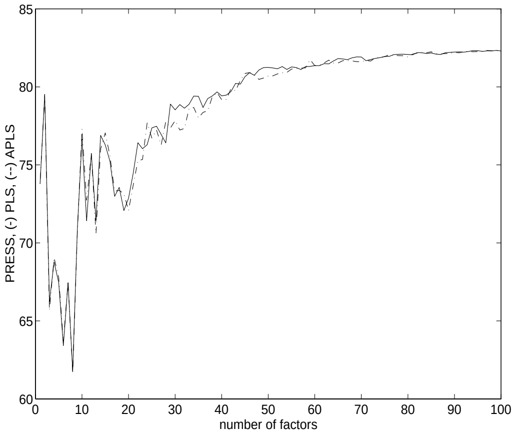

Figure 5: PRESS values of PLS and APLS, data set 2.

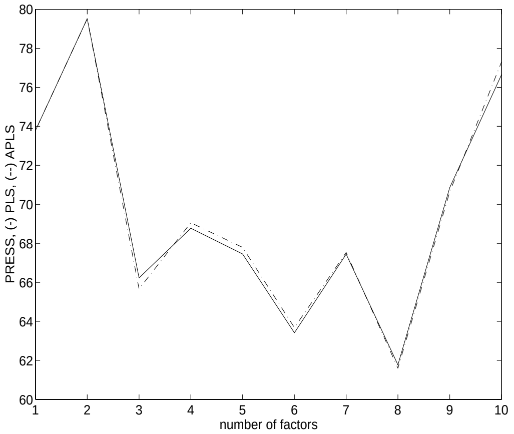

Figure 6: PRESS values of PLS and APLS for data set 2, around the selected factors.

As the third data set, we used a set of 30 combustible samples for which the NIR spectra over 3632 channels have been measured. Samples were reduced to contain only 363 measures by using every tenth response and then mean centered. 21 samples were utilized for calibrating (70% of the set) and the remaining 9 for validating. As the dependent variables, concentrations of three components were used for each sample.

In Figure 7 we show the performance of PLS and APLS for the first 100 factors.

A detailed view in the range of factor from 1 to 12 is presented in Figure 8.

As we can see, the results obtained with the APLS algorithm are very competitive against those of standard PLS. In Table 1 we summarize our empirical findings. First the minimum PRESS for both methods are nearly the same. Second, the chosen number of factors are similar: for the first data set the best choice is 32 for PLS and 34 for APLS, and for the second data set, 8 factors were chosen for both models.

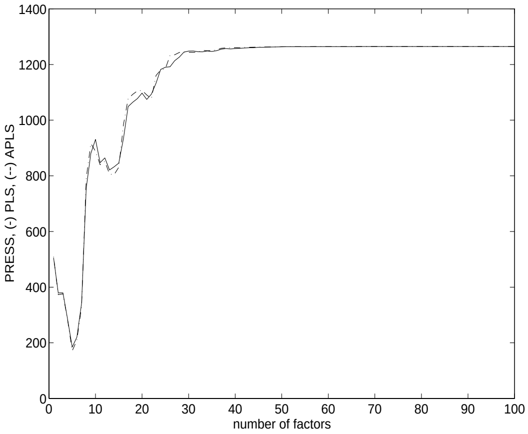

Figure 7: PRESS values of PLS and APLS, data set 3.

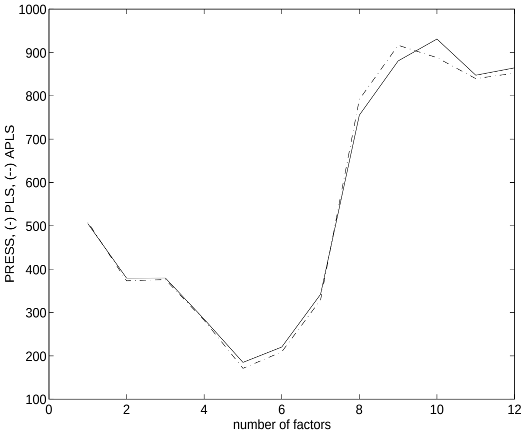

Figure 8: PRESS values of PLS and APLS for data set 3, around the selected factors.

6. Conclusions

We have presented APLS, an approximate algorithm for the standard Partial Least-Squares problem. Instead of computing the eigenvector of the matrix $X^\top YY^\top X$ in the case of 2 or more dependent variables, APLS approximate this result by decomposing the given matrix in a special way.

We have made experiments with three data sets commonly found in the chemometric area, and among the main characteristics of APLS, we can cite:

1. minimum PRESS found is similar to the one found by PLS;

2. similar number of factors chosen when compared to standard PLS;

3. improved running time, since we obtained a 30 % better time in our experiments.

Table 1: PRESS and Number of Chosen Factors for PLS and APLS.

Data Set	Algorithm	Optimal N. Factors	PRESS
1	PLS	32	6,12
	APLS	34	6,62
2	PLS	8	61,77
	APLS	8	61,60
3	PLS	5	184,79
	APLS	5	171,09

As we can see, APLS can be considered an alternative approach to PLS. Furthermore, the parallelization of APLS is straightforward [13], making it suitable for large data sets.

References

[1] P. Geladi and B. R. Kowalski. Partial least squares regression: A tutorial. *Analytica Chimica Acta*, 185:1–17, 1986.

[2] D. M. Haaland and E. V. Thomas. Partial least-squares methods for spectral analyses. 1. relation to other quantitative calibration methods and the extraction of qualitative information. *Analytical Chemistry*, 60(11):1193–1202, June 1988.

[3] D. M. Haaland and E. V. Thomas. Partial least-squares methods for spectral analyses. 2. application to simulated and glass spectral data. *Analytical Chemistry*, 60(11):1202–1208, June 1988.

[4] K. R. Beebe and B. R. Kowalski. An introduction to multivariate calibration and analysis. *Analytical Chemistry*, 59(17):1007–1017, September 1987.

[5] M. T. Alain Morineau, editor. *Les Méthodes PLS, Symposium International PLS'99*. Cisia-Ceresta, October 1999.

[6] NEURAP'98. *Time Series Forecasting throught Wavelets and a Mixture of Experts Models*, 1998.

[7] R. Milidiú, R. Machado, and R. Rentería. Time series forecasting throught wavelets and a mixture of experts models. *Neurocomputing*, 28:145–156, 1999.

[8] S. Haykin. *Neural Networks, a Comprehensive Foundation*. Prentice-Hall, 1999.

[9] J. H. Kalivas. Two data sets of near infrared spectra. *Chemometrics and Intelligent Laboratory Systems*, 37:255–259, 1997.

[10] Data set three, light gas oil (and diesel) fuels. http://www.dal.ca/ pdwentze/index.html.

[11] L. Lebart, A. Morineau, and M. Piron. *Statistique Exploratoire Multidimensionnelle*. Dunod, Paris, 1997.

[12] G. H. Golub and C. F. V. Loan. *Matrix Computations*. John Hopkins University Press, 1996.

[13] R. L. Milidiú and R. Rentería. Ppls: An efficent parallel algorithm for partial least-squares. In *12th Symposium on Computer Architecture and High Performance Computing*, São Pedro, SP, Brazil, October 2000.