LA-UR 96-3958

Los Alamos National Laboratory is operated by the University of California for the United States Department of Energy under contract W-7405-ENG-36

TITLE: **PARTIAL LEAST SQUARES, CONJUGATE GRADIENT AND
THE FISHER DISCRIMINANT**

AUTHOR(S): Vance Faber

SUBMITTED TO: External Distribution - Hard Copy

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

MASTER

By acceptance of this article, the publisher recognizes that the U.S. Government retains a nonexclusive royalty-free license to publish or reproduce the published form of this contribution or to allow others to do so, for U.S. Government purposes.

The Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy.

Los Alamos

Los Alamos National Laboratory
Los Alamos New Mexico 87545

DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.

DISCLAIMER

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, make any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

Partial Least Squares, Conjugate Gradient and the Fisher Discriminant

Vance Faber

1 Synopsis and Outline

The theory of multivariate regression has been extensively studied and is commonly used in many diverse scientific areas. A wide variety of techniques [1,2,3,7] are currently available for solving the problem of multivariate calibration. The volume of literature on this subject is so extensive that understanding which technique to apply can often be very confusing.

A common class of techniques for solving linear systems, and consequently applications of linear systems to multivariate analysis, are *iterative* methods. While common linear system solvers typically involve the factorization of the coefficient matrix A in solving the system $Ax = b$, this method can be impractical if A is large and sparse. Iterative methods such as Gauss-Seidel, SOR, Chebyshev semi-iterative, and related methods also often depend upon parameters that require calibration and which are sometimes hard to choose properly. An iterative method which surmounts many of these difficulties is the method of conjugate gradient. Algorithms of this type find solutions iteratively, by optimally calculating the next approximation from the residuals.

Partial Least Squares Regression (PLS) is a type of iterative method which has been introduced recently and whose development is attributed to Hermann Wold and Harald Martens. An exposition of the method is given in *Multivariate Calibration* [9], a text whose intended audience is analytical chemists. The main purpose of this paper is to show that PLS is an inefficient iterative method that combines two essentially standard conjugate gradient algorithms. The first, known as Craigs method, solves a least squares type minimization in the standard inner product, and this is what PLS uses in its training stage. Essentially, the normal equations

$$A^T A x = A^T b$$

corresponding to an over determined linear system $Ax = b$ are solved iteratively, at the k -th step solving the minimization

$$\min_{\tilde{x} \in V_k} \|x - \tilde{x}\| ,$$

where V_k is an iteratively computed linear space known as a Krylov space. That is, at the k -th step the projection of the least squares solution to the system on the Krylov space is found. A discussion of Krylov spaces is given in Section 3. The solution

$$\tilde{x} = \sum_i \beta_i w_i$$

is the expansion in the orthogonal basis vectors $\{w_i\}$ where orthogonality is with respect to the usual Euclidean inner product $\langle w_i, w_j \rangle = w_i^T \cdot w_j = \delta_{ij}$ where δ_{ij} is the delta function.

The second iterative method which we consider is known as the *Orthomin* algorithm. It takes a distinctly different strategy. In this case the minimization is performed on how well the range of the approximate solution approximates the target vector b . At the k -th step the minimization

$$\min_{\tilde{x} \in V_k} \|A\tilde{x} - b\|$$

is solved where the range of the solution is orthogonally expanded as

$$A\tilde{x} = \beta_0 Av_0 + \beta_1 Av_1 + \cdots + \beta_k Av_k .$$

In *Orthomin* the inner product is $\langle Av_i, Av_j \rangle = \beta_{ij}$.

The PLS algorithm works by expanding the solution in terms of what are called the spectral loadings by Martens and Naes, using the same inner product as Craigs method, except without the optimality. At the final step, however, the prediction stage, the least squares solution to the problem $Ax = b$ is solved by minimizing residuals as in the *Orthomin* algorithm. While it may certainly be of interest to the practitioner to adopt minimization in either of these inner products, a key point is that the end result of PLS may be obtained without Craigs method by simply implementing the significantly more direct and simpler *Orthomin* algorithm.

We begin in Sections 2 and 3 by reviewing the basics of linear regression and conjugate gradient iterative methods. In Sections 4 and 5 two of the most popular conjugate gradient methods are introduced, Craigs method and *Orthomin*. In Section 6 we show how PLS as written by Martens and Naes may be formulated as a conjugate gradient method. We present our main results in Section 7, demonstrating that the training and prediction stages of PLS have the properties claimed above. In the final section we summarize our findings. Various implementation issues and the connections of PLS with the Fisher linear discriminant are discussed in the appendices.

2 The Least Squares Approach

The standard m -variable linear regression model is an effort to solve the underlying over determined system $Ax = b$. It involves the dependent variable b and m independent variables numbered $A^{(1)}, A^{(2)}, \dots, A^{(m)}$ and may be expressed in matrix form as

$$b = Ax + e \tag{2.1}$$

where b is a $n \times 1$ column vector of observations of the dependent variable, and A is a $n \times m$ matrix that results from n observations of the m independent variables $A^{(1)}, A^{(2)}, \dots, A^{(m)}$. The column vector e is an $n \times 1$ column vector of residuals e_i . The $m \times 1$ column vector x contains the m regression coefficients $x^{(1)}, x^{(2)}, \dots, x^{(m)}$. The regression model (2.1) yields the expected value of $b = (b_1, \dots, b_n)$, given the fixed coordinate values of $A^{(i)} = (A^{(1i)}, A^{(2i)}, \dots, A^{(mi)})$, plus the error component. (The j -th regression coefficient $x^{(j)}$ measures the change in the mean value of b per unit change in $A^{(j)}$ holding all other independent variables constant.)

Standard assumptions in most least squares formulations are

1. The error term e has a multivariate gaussian distribution with expectation $E(e) = 0$,
2. $E(e \cdot e^T) = \sigma^2 I$, where I is the identity matrix and σ^2 is the constant variance of the residuals,
3. $Cov(e, A) = 0$, that the covariance between the A s and the residual terms is zero. This assumption is automatically satisfied if the A variables are non-stochastic so that the $n \times m$ data matrix A consists of fixed numbers, and finally,
4. The rank $r(A) = m$ where $m < n$. This assumption means that there are no linear relationships among the A variables, better known as no multicollinearity, stated otherwise, the population regression function should include only those variables that are not linear functions of some of the variables in the model.

When these four assumptions are satisfied then the least squares estimator, given below in (2.3), is in the class of minimum variance unbiased estimators, often called BLUE for Best Linear Unbiased Estimator. Under these assumptions, the least squares solution is also the maximum likelihood estimator of the mean.

The ordinary least squares procedure consists of choosing unknown parameters so that $e^T e = (b - Ax)^T (b - Ax)$ is as small as possible. Differentiating this last expression with respect to the unknowns

$x = \{x^{(i)}\}$, setting the resulting expressions equal to zero, and solving simultaneously yields, in matrix form, what are called the *normal* equations which may be written as

$$A^T A x = A^T b \quad (2.2)$$

If the inverse of the square matrix $A^T A$ exists, (and this is strongly dependent upon the assumptions 1-4 above as well as issues of numerical stability) then the unknown coefficients may be solved for as

$$x = (A^T A)^{-1} A^T b \quad (2.3)$$

Another possibility for a least squares solution is to let $x = A^T h$ and attempt to solve $AA^T h = b$. Now AA^T is a symmetric $n \times n$ matrix but its rank is m , and so this problem may not be solved exactly unless b is in the range of AA^T . If the vector x from this approach differs from the x given by the normal equations (2.2) then we may write $x = A^T h + t$ where t is orthogonal to the range of A^T . However, as any vector orthogonal to the range of A^T is necessarily in the null space of A , t adds nothing to the least squares solution of this problem. Thus from the least squares point of view, these two approaches are equivalent.

3 Conjugate Gradient Methods

The conjugate gradient method for the least squares solution of symmetric positive definite matrices has proven to be very effective, especially when combined with certain preconditioning techniques. Even when the matrix is nonsingular, but not positive definite extensions of the method continue to make it viable for the solution of linear regression problems. Preconditioning techniques can extend the solution class even further. The distinguishing characteristics of a conjugate gradient method are

1. It is a Krylov space method. That is, the solution x_k at any step k lies in the span of a certain subspace which is iteratively computed.
2. It has an optimality property over the Krylov space, which usually means convergence to an acceptable accuracy in a small number of iterations.
3. There are orthogonal direction vectors which span the Krylov space, and these are simultaneously computed.
4. The coefficients for updating the recursion are optimally chosen with respect to some inner product. A short recurrence relation makes the execution time per iteration short and memory requirements acceptable.
5. The method is parameter-free, that is, the user is not required to estimate any method parameters.
6. The roundoff error properties are acceptable.

We now explain how this type of method works, closely following Faber and Manteuffel [4,5]. Given the linear system $Cx = b$ and an initial guess x_0 at the solution, let the initial residual be denoted by $r_0 = b - Cx_0$. A *gradient method* is defined as a method in which

$$x_{i+1} = x_i + \sum_{j=0}^i \eta_{ij} r_j \quad (3.1)$$

where residual at step $j + 1$ is $r_j = b - Cx_j$, that is, at each step the partial solution is incremented by some linear combination of the previous residual vectors. The term gradient comes from the fact that if C is symmetric, $r = b - Cx$ is the gradient of the bilinear form

$$Q(x) = \frac{1}{2} \langle C^{-1}b - x, C(C^{-1}b - x) \rangle \quad ,$$

and if C is symmetric positive definite as well, then $Q(x)$ is minimized by $x = C^{-1}b$. Even when C is not symmetric but positive definite, we may still consider the iterative method of (3.1), as each is easily computable. To demonstrate this, we first show by induction that there is a polynomial of degree $\leq j$. Since $x_1 = x_0 + \eta_{00}r_0$ we may write $r_1 = (I - \eta_{00}C)r_0 = p_1(C)r_0$. In general,

$$r_{i+1} = r_i - \sum_{j=0}^i \eta_{ij}Cr_j = \left[(I - \eta_{ii}C)p_i(C) - C \sum_{j=0}^{i-1} \eta_{ij}p_j(C) \right] r_0$$

which we may define to be $p_{i+1}(C)r_0$, the required polynomial.

If $\eta_{ii} \neq 0$ for all i then the polynomial $p_i(z)$ is exactly of degree i and the residual vectors form a basis for the vector space generated by C and r_0 . This space is called a *Krylov* space and is defined as the span

$$V_i = \{r_0, Cr_0, \dots, C^i r_0\} = \{r_0, r_1, \dots, r_i\}.$$

From equation (3.1) we see that by a proper choice of the η_{ij} one can make $x_{i+1} - x_i$ an element of V_{i+1} .

Since the matrix C is of finite dimension N there must be some $l \leq N$ such that $V_l = V_{l+1}$. Let k be the least such l . Then the vectors $\{r_0, \dots, C^k r_0\}$ are linearly dependent and therefore there are constants $\alpha_1, \dots, \alpha_k$ such that

$$\sum_{i=0}^k \alpha_i C^i r_0 = 0. \quad (3.2)$$

If C is nonsingular, either $\alpha_0 \neq 0$ or a smaller k may be chosen. This is because if $\alpha_0 = 0$ then we may write (3.2) in the form $Cz = 0$ which has a non-zero solution only if C is non-singular. Then $r_0 = C(x - x_0)$ and the term $i = 0$ may be isolated. On the other hand if $\alpha_0 \neq 0$ the solution to the system $Cx = b$ is given by

$$x = x_0 - \frac{1}{\alpha_0} \sum_{i=1}^k \alpha_i C^{i-1} r_0.$$

Since the residual vectors span the Krylov space, the proper choice of the η_{ij} s in (3.1) will yield the solution. This approach in fact works even more generally. If C is singular, but has only simple zero eigenvalues, and the system is consistent, one can also obtain a solution.

The problem of course is choosing the η_{ij} s of (3.1) in a meaningful way. One way to do this is to enforce an optimality condition. The Krylov space is constructed to allow optimal choice of update directions. Let $e_i = C^{-1}b - x_i = x - x_i$ be the error vector. Let $\|x\| = \langle x, x \rangle^{1/2}$. We would like to choose x_i so that $\|e_i\|$ is minimized over all possible iterations of form (3.1). That is, at each step i let $x_{i+1} = x_i + \beta_i p_i$ where $p_i \in V_i$. We want to choose $\beta_i p_i$ so that $\|e_{i+1}\|$ is as small as possible. Since

$$x_{i+1} = x_0 + \sum_{j=0}^i \beta_j p_j$$

we have

$$e_{i+1} = e_0 - \sum_{j=0}^i \beta_j p_j.$$

Now take the inner product of both sides of the previous equation to get

$$\|e_{i+1}\|^2 = \langle e_0, e_0 \rangle - \left\langle e_0, \sum_{j=0}^i \beta_j p_j \right\rangle - \left\langle \sum_{j=0}^i \beta_j p_j, e_0 \right\rangle + \left\langle \sum_{j=0}^i \beta_j p_j, \sum_{j=0}^i \beta_j p_j \right\rangle.$$

To minimize the error, assume a complex solution $\beta_j = x_j + iy_j$, and take the derivative with respect to each $x_j y_j$. Setting these latter expressions equal to zero one obtains the linear system

$$\begin{bmatrix} \langle p_0, p_0 \rangle & \langle p_1, p_0 \rangle & \cdots & \langle p_i, p_0 \rangle \\ \langle p_0, p_1 \rangle & \cdot & \cdot & \cdot \\ \vdots & \vdots & \vdots & \vdots \\ \langle p_i, p_0 \rangle & \cdot & \cdot & \langle p_i, p_{i-1} \rangle \\ \langle p_0, p_i \rangle & \cdot & \cdot & \langle p_i, p_i \rangle \end{bmatrix} \begin{bmatrix} \beta_0 \\ \cdot \\ \cdot \\ \beta_{i-1} \\ \beta_i \end{bmatrix} = \begin{bmatrix} \langle e_0, p_0 \rangle \\ \cdot \\ \cdot \\ \langle e_0, p_{i-1} \rangle \\ \langle e_0, p_i \rangle \end{bmatrix}.$$

Since this is true for every i and in particular for $i-1$, we have

$$\beta_i \begin{bmatrix} \langle p_0, p_i \rangle \\ \vdots \\ \langle p_{i-1}, p_i \rangle \end{bmatrix} = 0, \quad \beta_i = \frac{\langle e_0, p_i \rangle}{\langle p_i, p_i \rangle}.$$

If $\beta_i \neq 0$, then $\langle p_j, p_i \rangle = 0$. If $\beta_i = 0$ then $\|e_{i+1}\| = \|e_i\|$. If $V_i = V_{i+1}$ we have seen above that V_i yields the solution, which implies that $e_{i+1} = 0$. If, on the other hand, $V_i \neq V_{i+1}$, there is a unique up-to-scale vector $p_i \in V_{i+1}$ such that $\langle p_j, p_i \rangle = 0$ for every $j \in V_i$. To summarize, if each $x_{i+1} = x_0 + y_i$ is chosen such that $\|e_{i+1}\|$ is optimal over all possible $y_i \in V_i$ then $x_{i+1} - x_i = \beta_i p_i$, where p_i is the unique (up-to-scale) vector $p_i \in V_{i+1}$ such that $\langle p_i, z \rangle = 0$ for every $z \in V_i$ and

$$\beta_i = \frac{\langle e_i + \sum_{j=0}^{i-1} \beta_j p_j, p_i \rangle}{\langle p_i, p_i \rangle} = \frac{\langle e_i, p_i \rangle}{\langle p_i, p_i \rangle}.$$

Since the p_i s are conjugate with respect to the inner product, we say that the method is a *conjugate gradient method*. Equation (3.3) involves the unknown quantity e_i , and so in order for the algorithm to be viable, the inner product must be chosen so that the coefficient β_i is computable.

Another issue which is central to the implementation of a conjugate gradient method is the recursive calculation of the p_i 's which we now discuss. From the above discussion, we know that since $p_i \in V_{i+1}$ and p_i is orthogonal to V_i , that $p_i = p_i(C)p_0$, where $p_i(z)$ is a polynomial of exact degree i . Thus, Cp_i is of exact degree $i+1$ and so $V_{i+2} = \{Cp_i, p_0, \dots, p_i\}$. The vector p_{i+1} can be computed as

$$p_{i+1} = Cp_i - \sum_{j=0}^i \alpha_{ij} p_j.$$

The α_{ij} s are then uniquely determined by the conditions,

$$\langle p_{i+1}, p_j \rangle = 0, \quad j = 0, \dots, i,$$

which yields

$$\alpha_{ij} = \frac{\langle Cp_i, p_j \rangle}{\langle p_j, p_j \rangle}, \quad j = 0, \dots, i.$$

Now if C is self-adjoint with respect to this inner product, we have

$$\alpha_{ij} = \frac{\langle p_i, C^T p_j \rangle}{\langle p_j, p_j \rangle} = \frac{\langle p_i, Cp_j \rangle}{\langle p_j, p_j \rangle}.$$

Since $p_j \in V_{j+1}$ we have $Cp_j \in V_{j+2}$. Since p_i is orthogonal to V_i , we also have $\alpha_{ij} = 0$ for $j < i-1$. Thus

$$p_{i+1} = Cp_i - \alpha_{i,i} p_i - \alpha_{i,i-1} p_{i-1}.$$

Equation (3.4) referred to as a *three-term recursion* for computing the update directions. More sophisticated methods exist involving s -term recurrences and these methods have been completely classified in the fascinating paper [4]. The most commonly used form of the three-term conjugate gradient method applied to symmetric positive definite C has general steps given after initialization as

$$\begin{aligned}
x_{i+1} &= x_i + \beta_i p_i, \\
r_{i+1} &= r_i - \beta_i C p_i, \\
p_{i+1} &= r_{i+1} + \alpha_i p_i.
\end{aligned}$$

Notice that $r_i = p_i - \alpha_{i-1} p_{i-1}$ so that

$$r_{i+1} = p_i - \alpha_{i-1} p_{i-1} - \beta_i C p_i.$$

Substituting for r_{i+1} above yields

$$p_{i+1} = -\beta_i C p_i + (1 + \alpha_i) p_i - \alpha_{i-1} p_{i-1}$$

which is the same as equation (3.4).

4 Craigs Method for Conjugate Gradient

One of the most commonly employed conjugate gradient methods is known Craigs method which we now introduce. The method enjoys all the properties (1-6) described at the beginning of the previous section and is often taken as the benchmark for other conjugate gradient methods. We also give the algorithm in the case when $A^T A$ has been precalculated.

Throughout, we wish to approximate the solution to the linear system $Ax = b$. Let r denote the residual vector $r = Ax - b$. In the algorithm given below, the subscripted variable x_i denotes the i -th approximation to the least squares solution. The algorithm has the following steps:

Algorithm 1: *Craigs Method.*

Initialize $x_0 = 0$, $r_0 = b$, $p_0 = A^T r_0 = A^T b$. Iterate for $1 \leq i \leq m$,

1. $x_{i+1} = x_i + \beta_i p_i$,
2. $r_{i+1} = r_i - \beta_i A p_i$,
3. $p_{i+1} = A^T r_{i+1} + \alpha_{i+1} p_i$

where $\beta_i = \langle r_i, r_i \rangle / \langle p_i, p_i \rangle$ and $\alpha_i = \langle r_i, r_i \rangle / \langle r_{i-1}, r_{i-1} \rangle$. and .

There are several important facts about Craigs method. The first is that it is a Krylov space method to solve the normal equations $A^T A x = A^T b$. This means that the solution estimate x_i is always contained in the space spanned by the vectors

$$V_i \{A^T b, (A^T A) A^T b, \dots, (A^T A)^i A^T b\}.$$

As discussed in the previous section we have $V_i \subseteq V_{i+1}$ for all i and the vectors p_i , called the update directions, defined by Craigs method form a basis for the subspace V_i which is computed iteratively along with the solution estimate: thus we may write $V_i = \{p_1, \dots, p_i\}$. A second point of interest about method is that the update directions p_i are orthogonal in the *usual* inner product and therefore $\langle p_i, p_j \rangle = p_i^T p_j = \delta_{ij}$ where δ_{ij} is the delta function. At the k -th step of Craigs method we are solving the minimization

$$\min_{\tilde{x} \in V_k} \|x - \tilde{x}\|$$

where $\tilde{x} \in \sum \beta_i p_i$ and $\langle p_i, p_j \rangle = \delta_{ij}$. A remarkable and non-trivial property of Craigs method is that we are iteratively able to find the closest solution in the current Krylov space to the least squares solution of the system $Ax = b$ without explicitly knowing the least squares solution. A justification for this fact is beyond the scope of this paper but can be found in [1] along with characterization results. A final point that makes Craigs method very efficient in practice is that, like all conjugate gradient methods, the basis vectors p_i satisfy a three term recurrence.

In the case where the symmetric positive definite matrix $A^T A$ has been precalculated, such as may be the case in reading very large data files from disk, the algorithm takes the following standard form [1].

Algorithm 2: *Craigs Method* with precalculated $C = A^T A$.

Initialize $x_0 = 0, p_0 = s_0 = b$.

Iterate for $1 \leq i \leq m$,

1. $x_{i+1} = x_i + \beta_i p_i$,
2. $s_{i+1} = s_i - \beta_i C p_i$,
3. $p_{i+1} = s_{i+1} + \alpha_{i+1} p_i$,

where $\beta_i = \rho_i / \langle p_i, p_i \rangle$ and $\alpha_i = \rho_i / \rho_{i-1}$, where ρ_i is calculated by the recursion $\rho_{i+1} = \rho_i - 2\beta_i \langle s_i, p_i \rangle + \beta_i^2 \langle p_i, A^T A p_i \rangle$ with $\rho_0 = \langle b, b \rangle$.

5 The *Orthomin* Algorithm for Conjugate Gradient

Another common conjugate gradient algorithm, which has a very similar form to Craigs method, is called *Orthomin* and is given below.

Algorithm 3: *Orthomin*

Initialize $x_0 = 0, r_0 = b, p_0 = A^T r_0 = A^T b$

Iterate for $1 \leq i \leq m$,

1. $x_{i+1} = x_i + \beta_i p_i$,
2. $r_{i+1} = r_i - \beta_i A p_i$,
3. $p_{i+1} = A^T r_{i+1} + \alpha_{i+1} p_i$,

where $\beta_i = \langle A^T r_i, A^T r_i \rangle / \langle A p_i, A p_i \rangle$ and $\alpha_i = \langle A^T r_i, A^T r_i \rangle / \langle A^T r_{i-1}, A^T r_{i-1} \rangle$.

As with Craigs method, this algorithm is a Krylov space method to solve the normal equations $A^T A x = A^T b$. As discussed above, the solution estimate is contained in the space spanned by the set of vectors $\{A^T b, (A^T A) A^T b, \dots, (A^T A)^i A^T b\}$. The second point is that the update directions p_i are orthogonal in an inner product, although this inner product is different from the ordinary inner product used by Craigs method. At the k -th step of the *Orthomin* algorithm we minimize

$$\min_{\tilde{x} \in V_k} \| A \tilde{x} - b \|$$

and the solution

$$A \tilde{x} = \beta_0 A p_0 + \beta_1 A p_1 + \dots + \beta_k A p_k$$

where the inner product is now in $C = A^T A$ so that $\langle A p_i, A p_j \rangle = \langle C p_i, p_j \rangle = \delta_{ij}$.

The algorithm *Orthomin* with precalculated $A^T A$ is identical to the algorithm for Craigs method with precalculated $A^T A$ except that the update coefficients are calculated as

$$\alpha_i = \langle s_i, s_i \rangle / \langle s_{i-1}, s_{i-1} \rangle ,$$

$$\beta_i = \langle s_i, s_i \rangle / \langle A^T A p_i, p_i \rangle .$$

A note on stopping criteria.

Since there is no x which satisfies $Ax = b$ in an overdetermined system, it is of little value to check the norm of the residual $\| b - Ax_i \|$, although in fact in the *Orthomin* algorithm this value steadily decreases to zero. As a stopping criterion, a better indication of convergence is the quantity $\| A^T r_i \| = \| A^T b - A^T A x_i \|$ the residual of the normal equations. Since both *Orthomin* and Craigs method calculate $A^T r_i$ during the iteration, it is a simple matter to calculate $\langle A^T r_i, A^T r_i \rangle$. This quantity is not strictly decreasing but does tend to zero during the iteration.

6 PLS as an Iterative Method

In this section we give a detailed explanation of how partial least squares regression may be interpreted as an iterative conjugate gradient method. We take the algorithm PLSR for one y-variable given in Frame 3.4 page 121 of the text *Multivariate Calibration* [9] by H. Martens and T. Naes. We shall discuss each step in this frame in detail referring to the same numbering scheme.

C.1 This step scales the independent variables by subtracting the mean. Mean centering is important in order to insure that the least squares (LS) solution includes the correct constant term. This step is essential if one is to solve least squares problems in affine spaces.

C 2.1 Martens and Naes (MN) define the *loading weight vector*

$$w_{i+1} = X_i^T y_i / \| X_i^T y_i \parallel$$

In our notation this will be denoted as

$$w_i = A_i^T b_i / \| A_i^T b_i \parallel$$

with $y_i = b_i$, $X_i = A_i$, and where the change of w subscript is intentional.

C 2.2 MN define the *factor scores*

$$t_{i+1} = X_i w_{i+1}$$

as the projection of X_i on w_{i+1} . We do not require this quantity directly, but use it implicitly in the next step.

C 2.3 MN estimate the *spectral loadings* (we have switched to capital letters)

$$P_{i+1} = X_i^T t_{i+1} / \| t_{i+1} \parallel^2$$

which may be equivalently written as

$$X_i^T X_i w_{i+1} / \| X_i w_{i+1} \parallel^2$$

In our notation this will be denoted as

$$P_i = A_i^T A_i w_i / \| A_i w_i \parallel^2 \quad (6.1)$$

C 2.4 MN estimate the *chemical loading*

$$q_{i+1} = y_i^T t_{i+1} / \| t_{i+1} \parallel^2$$

In our method this will be denoted as

$$q_i = b_i^T A_i w_i / \| A_i w_i \parallel^2 \quad (6.2)$$

C 2.5 At this point MN give their update equations. In our notation these are given by

$$\begin{aligned} A_{i+1} &= (I - u_{i+1} u_{i+1}^T) A_i \\ u_{i+1} &= A_i A_i^T b_i / \| A_i A_i^T b_i \parallel \end{aligned}$$

and

$$\begin{aligned} b_{i+1} &= b_i - a_i A_i A_i^T b_i \\ a_i &= \| A_i^T b_i \parallel^2 / \| A_i A_i^T b_i \parallel^2 \end{aligned}$$

These equations will form the starting point of our proof that PLS is in fact a Krylov space method, given in the next section.

C 3 In MN this step asks to check the accuracy of the solution. This step is similar in any iterative method and usually the number of factors a is predetermined.

C 4 At this point the iteration has terminated and PLS has calculated two sets of column vectors $\{w_0, w_1, \dots, w_{a-1}\}$ and $\{P_0, P_1, \dots, P_{a-1}\}$, and two sets of scalar coefficients, $\{q_0, q_1, \dots, q_{a-1}\}$ and $\{a_0, a_1, \dots, a_{a-1}\}$. These latter coefficients are explicitly given in the language of MN as $a_{i+1} = 1 / \|t_{i+1}\|^2$. Some authors [6] suggest that the PLS approximation to the solution is best given as

$$\tilde{x} = \sum_{i=0}^{a-1} a_i q_i w_i = \sum_{i=1}^{a-1} a_i A_i^T A_i b_i \quad (6.3)$$

however, the authors MN try a different approach. They let W be the matrix with the vectors w_i as columns, P be the matrix with P_i as columns, and q be the column vector having q_i as entries. Then MN take the approximation to the solution, the PLS prediction step, to be

$$\tilde{x} = W(P^T W)^{-1} q \quad (6.4)$$

We shall show in the next section that the w_i are orthogonal in the ordinary inner product, but that the P_i are not. We shall also show that (6.3) is not the vector closest to x in the space spanned by the w_i , but that $\tilde{x}$ in (6.4) is the minimum of the normal equations

$$\min \|A^T A \tilde{x} - A^T b\| = \min \|A \tilde{x} - b\|$$

with $\tilde{x}$ in the space spanned by the w_i .

If the solution to the PLS method is taken as in (6.3), then the iteration resembles Craigs method except that the coefficients have not been chosen optimally. If, on the other hand, the solution is chosen as in (6.4), then we have solved the same problem as *Orthomin*, but we have orthogonalized in the wrong space so a matrix inverse is required. This can cause two undesirable side effects. The first is that we cannot calculate how accurate our solution is at each step so we cannot determine a stopping criterion very easily. The second point is that a matrix inverse is numerically unstable and inefficient. The conjugate gradient methods avoid these difficulties. To understand PLS as the optimization carried out in *Orthomin*, we observe that as remarked above we must solve the minimization

$$\min_{x \in V_m} \|Ax - b\|$$

where $x \in V_m = \langle A^T b, CA^T b, \dots, C^m A^T b \rangle$. Form a matrix U with columns $C^i A^T b$ for $1 \leq i \leq m$. As we are approximating x by Uy and so we solve for y in the problem

$$\min \|AUy - b\|$$

As before, the least squares solution is given by

$$y = (U^T A^T A U)^{-1} U^T A^T b$$

and the approximation to the solution x is given by

$$x \approx U(U^T A^T A U)^{-1} U^T A^T b$$

As previously remarked, performing the matrix inverse has undesirable side effects.

7 Conjugate Gradient and PLS

In this section we prove the necessary facts to demonstrate that PLS is a conjugate gradient type iteration scheme for least squares. Let us first recall our equivalent version of PLS in update form from Section 6.

1. $u_{i+1} = A_i A_i^T b_i / \|A_i A_i^T b_i\|,$
2. $a_i = \|A_i^T b_i\|^2 / \|A_i A_i^T b_i\|^2,$
3. $A_{i+1} = (I - u_{i+1} u_{i+1}^T) A_i,$

$$4. b_{i+1} = b_i - a_i A_i A_i^T b_i.$$

There are several points to note. First, recall that we have defined $w_i = A_i^T b_i / \|A_i^T b_i\|$. In what follows, we let $C = A^T A$ and V_i be the space spanned by $\{A^T b, CA^T b, \dots, C^i A^T b\}$. We also assume that p_i is a sequence of vectors which have been constructed so that for each j , $\{p_0, \dots, p_j\}$ is an orthogonal basis for the Krylov space V_j . The directions $p_i / \|p_i\|$ are then unique. We shall also make use of the fact that Cp_i is a linear combination only of p_{i-1}, p_i, p_{i+1} (The 3-term recursion mentioned in Section 3.)

Theorem 1. *The vectors $A_i^T b_i$ and $A_j^T b_j$ are orthogonal for $i \neq j$.*

The proof will require a few lemmas.

Lemma 1. *For every i , u_{i+1} and b_{i+1} are orthogonal.*

Proof. Note that

$$u_{i+1}^T b_{i+1} = b_i^T A_i A_i^T (b_i - a_i A_i A_i^T b_i) / \|A_i A_i^T b_i\|.$$

Multiplying the right hand side by $\|A_i A_i^T b_i\| / \|A_i A_i^T b_i\|$, and factoring $\|A_i A_i^T b_i\|$, we obtain

$$\|A_i A_i^T b_i\| \left(\frac{\|A_i^T b_i\|^2}{\|A_i A_i^T b_i\|^2} - a_i \right) = 0.$$

Lemma 2. *For every i , $A_{i+1}^T b_{i+1} = A_i^T b_{i+1}$.*

Proof. We have

$$\begin{aligned} A_{i+1}^T b_{i+1} &= A_i^T (I - u_{i+1} u_{i+1}^T) b_{i+1} \\ &= A_i^T b_{i+1} - 0 \text{ by Lemma 1,} \\ &= A_i^T b_{i+1} \end{aligned}$$

Lemma 3. *For every i , $A_{i+1} A_i^T b_i = 0$.*

Proof. Notice that

$$A_i A_i^T b_i = u_{i+1} \|A_i A_i^T b_i\|.$$

We then compute,

$$\begin{aligned} &= (I - u_{i+1} u_{i+1}^T) u_{i+1} \|A_i A_i^T b_i\| \\ &= u_{i+1} \|A_i A_i^T b_i\| - u_{i+1} \|A_i A_i^T b_i\| = 0. \end{aligned}$$

Lemma 4. *For every $j > i$, $A_j A_i^T b_i = 0$.*

Proof. By induction, we have

$$A_{j+1} A_i^T b_i = (I - u_{j+1} u_{j+1}^T) A_j A_i^T b_i = 0.$$

We may now complete the proof of Theorem 1.

Proof of Theorem 1. Calculate by applying Lemma 4

$$\langle A_i^T b_i, A_j^T b_j \rangle = \langle A_j A_i^T b_i, b_j \rangle = 0 \text{ for } j > i.$$

and the theorem is proved.

Theorem 2. *The space spanned by $\{A_0^T b_0, \dots, A_i^T b_i\}$ is the Krylov space V_i .*

Proof. The first iterate $A_0^T b_0$ is just $A^T b$. The idea is to prove the theorem by induction by first computing the action of the rank 1 matrix $u_i u_i^T$. We notice that $u_i u_i^T v$ is in the direction of u_i for any vector v , as $u_i^T v$ is a scalar. Thus if u_i is in some Krylov space, then we infer that $u_i u_i^T v$ is in the same Krylov space. Also note that

$$\begin{aligned}
(I - u_{i+1}u_{i+1}^T)(I - u_{i+1}u_{i+1}^T) &= I - 2u_{i+1}u_{i+1}^T + u_{i+1}u_{i+1}^T u_{i+1}u_{i+1}^T \\
&= I - u_{i+1}u_{i+1}^T .
\end{aligned}$$

Assume by induction that $A_i^T b_i \in V_i$ and that $(A_i^T A_i)V_k \in V_i$ for $t = \max(k+1, i)$ for all k . We check whether these statements remain true for $i+1$. Compute

$$\begin{aligned}
A_{i+1}^T b_{i+1} &= A_i^T b_{i+1} \text{ by Lemma 2,} \\
&= A_i^T (b_i - a_i A_i A_i^T b_i) \text{ by update rule 1,} \\
&= A_i^T b_i - a_i (A_i^T A_i) A_i^T b_i ,
\end{aligned}$$

which is in V_t by induction, where $t = \max(i, i+1) = i+1$.

Now let p be any polynomial of degree k . Then $p(C)A^T b$ is a typical element of V_k where $C = A^T A$. Calculate

$$\begin{aligned}
A_{i+1}^T A_{i+1} p(C) A^T b &= A_i^T (I - u_{i+1}u_{i+1}^T)^2 A_i p(C) A^T b \\
&= A_i^T I - u_{i+1}u_{i+1}^T A_i p(C) A^T b \\
&= A_i^T A_i p(C) A^T b - A_i^T u_{i+1} [u_{i+1}^T A_i p(C) A^T b] .
\end{aligned} \tag{7.1}$$

Clearly $A_i^T A_i p(C) A^T b$ is in V_t for $t = \max(k+1, i)$, by induction. On the other hand, the term

$$\begin{aligned}
A_i^T u_{i+1} &= A_i^T [A_i A_i^T b_i / \|A_i A_i^T b_i\|] \\
&= [1 / \|A_i A_i^T b_i\|] (A_i^T A_i) A_i^T b_i ,
\end{aligned}$$

which is in V_{i+1} by the two induction hypothesis. Thus $A_{i+1}^T A_{i+1} p(C) A^T b$ is in V_t where $t = \max(k+1, i+1)$ which is what we needed to show to continue the induction. This completes the proof of Theorem 2.

Corollary 1. For all j , we have $p_j / \|p_j\| = A_j^T b_j / \|A_j^T b_j\| = w_j$.

Proof. This follows easily from the two theorems taken together and the uniqueness of the directions p_j .

The formula that was proved by induction in the theorem above is important enough to warrant its own lemma.

Lemma 5. For all i and k , $(A_i^T A_i)V_k$ is contained in V_t for $t = \max(k+1, i)$.

Lemma 6. For any vector p ,

$$A_j^T A_j p = A_{j-1}^T A_{j-1} p - d A_{j-1}^T A_{j-1} p_{j-1} \tag{7.2}$$

where

$$d = \langle A_{j-1}^T A_{j-1} p_{j-1}, p \rangle / \langle A_{j-1} p_{j-1}, A_{j-1} p_{j-1} \rangle .$$

Proof. This is an easy application of equation (7.1) and the fact that p_i and $A_i^T b_i$ are in the same direction.

Lemma 7. For all i ,

$$\langle A_j^T A_j p_i, p_k \rangle = \langle C p_i, p_k \rangle$$

when $j < k$, where $C = A^T A$.

Proof. We use equation (7.2) to write

$$A_j^T A_j p = A_{j-1}^T A_{j-1} p - d A_{j-1}^T A_{j-1} p_{j-1} .$$

Now we prove the statement of the theorem for fixed k by induction on i, j . Consider equation (7.2) with $p = p_i$. By Lemma 5 the second term on the right is in V_j and so is orthogonal to p_k when $k > j$. When we form the inner product with p_k , the first term on the right is

$$\langle Cp_i, p_k \rangle$$

by induction. This proves the Lemma.

Now we must deal with the update scale factor a_i . We handle this with Theorem 3. First we require a technical lemma.

Lemma 8. Let $d_i = \langle A_i p_i, A_i p_i \rangle$ then for every i ,

$$d_i = \langle Cp_i, p_i \rangle - \langle Cp_{i-1}, p_i \rangle^2 / d_{i-1} .$$

Proof. We evaluate d_i with the use of equation (7.2),

$$\begin{aligned} \langle A_i p_i, A_i p_i \rangle &= \langle A_i^T A_i p_i, p_i \rangle \\ &= \langle A_{i-1}^T A_{i-1} p_i, p_i \rangle - d \langle A_{i-1}^T A_{i-1} p_{i-1}, p_i \rangle \\ &= \langle Cp_i, p_i \rangle - d \langle Cp_{i-1}, p_i \rangle \text{ by Lemma 7} . \end{aligned}$$

Now with $p = p_i$ we have

$$\begin{aligned} d &= \langle A_{i-1}^T A_{i-1} p_{i-1}, p_i \rangle / \langle A_{i-1} p_{i-1}, A_{i-1} p_{i-1} \rangle \\ &= \langle Cp_{i-1}, p_i \rangle / \langle A_{i-1} p_{i-1}, A_{i-1} p_{i-1} \rangle , \\ &= \langle Cp_{i-1}, p_i \rangle / d - i - 1 . \end{aligned}$$

with the aid of Lemma 7 again. This proves the lemma.

Theorem 3. The update scale factor a_i can be written

$$a_i \langle p_i, p_i \rangle / d_i .$$

Proof. Recall that

$$a_i = \| A_i^T b_i \|^2 / \| A_i A_i^T b_i \|^2$$

and since we have shown that $A_i^T b_i$ is in the same direction as p_i , we have

$$a_i = \| p_i \|^2 / \| A_i p_i \|^2 = \| p_i \|^2 / d_i$$

Lemma 9. For every k ,

$$A_k^T A_k p_k = Cp_k - \langle Cp_{k-1}, p_k \rangle A_{k-1}^T A_{k-1} p_{k-1} / d_{k-1} .$$

Proof. The strategy is to show that both sides of the stated equation are equal upon taking inner products with all with all p_j . This sufficient as the p_j form a basis for the Krylov space. For j not equal to $k-1, k$ or $k+1$, Lemma 7 can be used in conjunction with the 3-term recursion for Cp_j to show that both sides of the relevant equation are zero.

With p_{k+1} we use Lemma 7 to get

$$-\langle A_k^T A_k p_k, p_{k+1} \rangle = \langle Cp_k, p_{k+1} \rangle ,$$

while the term on the far right is zero because of the 3-term recursion. With p_k , then by Lemma 8 the left hand side of the equation can be written as

$$\langle A_k^T A_k p_k, p_k \rangle = d_k = \langle Cp_k, p_k \rangle - \langle Cp_{k-1}, p_k \rangle^2 / d_{k-1} .$$

The far right term is

$$\langle Cp_{k-1}, p_k \rangle \langle A_{k-1}^T A_{k-1} p_{k-1}, p_k \rangle / d_{k-1} = \langle Cp_{k-1}, p_k \rangle^2 / d_{k-1}$$

by Lemma 7. With p_{k-1} ,

$$\langle A_k^T A_k p_k, p_{k-1} \rangle = 0$$

since $A_k p_{k-1} = 0$ by Lemma 3. The right hand side can be computed as

$$\begin{aligned} \langle Cp_k, p_{k-1} \rangle &= \langle Cp_{k-1}, p_k \rangle \langle A_{k-1}^T A_{k-1} p_{k-1}, p_{k-1} \rangle / d_{k-1} \\ &= \langle Cp_k, p_{k-1} \rangle - \langle Cp_{k-1}, p_k \rangle = 0 \end{aligned}$$

as well. This concludes the proof.

Theorem 4. *The vector spaces generated by $\{A_i^T A_i p_i | i < k\}$ and $\{Cp_i | i < k\}$ respectively, are identical.*

Proof. This follows easily by using Lemma 9 and induction.

In order to explain step C4 of Martens and Naes, we recall the definitions of P_k and q_k in Section 6 as equation (6.1) and (6.2)

$$P_k = A_k^T A_k w_k / \|A_k w_k\|^2,$$

$$q_k = \langle A_k^T b_k, w_k \rangle / \|A_k w_k\|^2.$$

Using this notation, we can rewrite (with a little algebra) Lemma 9 as

$$Cw_k = \sigma_k P_k + \tau_{k-1} P_{k-1}, \quad (7.3)$$

$$\sigma_k = \|A_k w_k\|^2 \quad (7.4)$$

$$\tau_k = \langle Cp_k, p_{k+1} \rangle \|p_k\| \sigma_k / (d_k \|p_{k+1}\|)$$

Lemma 10. *For any $k > 0$,*

$$\sigma_k q_k + \tau_{k-1} q_{k-1} = 0.$$

Proof. We compute

$$\sigma_k q_k + \tau_{k-1} q_{k-1} = \langle A_k^T b_k, w_k \rangle + \langle Cp_{k-1}, p_k \rangle \|p_{k-1}\| \langle A_{k-1}^T b_{k-1}, w_{k-1} \rangle / (d_{k-1} \|p_k\|).$$

Recall the following relation from the proof of Theorem 2:

$$A_k b_k = A_{k-1}^T b_{k-1} - a_{k-1} A_{k-1}^T A_{k-1} A_{k-1}^T b_{k-1} - k - 1.$$

Substituting this into the first inner product and using the fact that $\langle A_{k-1}^T b_{k-1}, w_k \rangle = 0$, the right hand side becomes

$$-a_{k-1} \langle A_{k-1}^T A_{k-1} A_{k-1}^T b_{k-1}, w_k \rangle + \langle Cp_{k-1}, p_k \rangle \|p_{k-1}\| \langle A_{k-1}^T b_{k-1} - k - 1, w_{k-1} \rangle / (d_{k-1} \|p_k\|)$$

which is

$$- \|p_{k-1}\|^2 \langle A_{k-1}^T A_{k-1} A_{k-1}^T b_{k-1}, w_k \rangle / d_{k-1} + \langle Cp_{k-1}, p_k \rangle \|p_{k-1}\|^2 \langle A_{k-1}^T b_{k-1}, w_{k-1} \rangle / (d_{k-1} \|p_k\| \|p_{k-1}\|)$$

by Theorem 3. But now factoring $\|p_{k-1}\|^2 / d_{k-1}$, and converting $\langle Cp_{k-1} / \|p_{k-1}\|, p_k / \|p_k\| \rangle$ the remaining expression is

$$\begin{aligned}
& - \langle A_{k-1}^T A_{k-1} A_{k-1}^T b_{k-1}, w_k \rangle + \langle C w_{k-1}, w_k \rangle \langle A_{k-1}^T b_{k-1}, w_{k-1} \rangle \\
& = - \| A_{k-1}^T b_{k-1} \| \langle C w_{k-1}, w_k \rangle + \langle C w_{k-1}, w_k \rangle \| A_{k-1}^T b_{k-1} \| \langle w_{k-1}, w_{k-1} \rangle \\
& = 0,
\end{aligned}$$

where the second equality employs Lemma 7. This completes the proof.

Lemma 11. For every $k > 0$,

$$\| A_k^T b_k \| = - \| A_{k-1}^T b_{k-1} \| (C p_{k-1}, p_k) \| p_{k-1} \| / (d_{k-1} \| p_k \|)$$

Proof. Substitute $A_k^T b_k = \| A_k^T b_k \| w_k$ into

$$\langle A_k^T b_k, w_k \rangle + \langle C p_{k-1}, p_k \rangle \| p_{k-1} \| \langle A_{k-1}^T b_{k-1}, w_{k-1} \rangle / (d_{k-1} \| p_k \|) = 0$$

from the proof of Lemma 10.

Our final goal in this section is to show that the step C4, the prediction stage of PLS given by Martens and Naes, forms the solution to the problem

$$\min_{x \in V_m} \| Ax - b \|^2$$

where V_m is the Krylov space spanned by $\{p_0, p_1, \dots, p_m\}$. First we find the solution to (7.5). Since V_m is also the space spanned by $\{w_0, w_1, \dots, w_m\}$, we can write $x = Wy$ and solve for y . The one is then given as

$$y = (W^T A^T A W)^{-1} W^T (A^T b) = (W^T C W)^{-1} W^T (A^T b)$$

and the value for x thus obtained is

$$x = W(W^T C W)^{-1} W^T A^T b.$$

The matrix CW has the column vectors $Cw_0, Cw_1, \dots, Cw_m$. By equation (7.3)

$$CW = PM$$

where M is the band matrix

$$M = \begin{pmatrix} \sigma_0 & \tau_0 & & 0 \\ & \sigma_1 & \tau_1 & \\ & & \dots & \dots \\ & 0 & & \sigma_{n-1} & \tau_{n-1} \end{pmatrix}.$$

Since we have

$$\begin{aligned}
x &= W(M^T P^T W)^{-1} W^T A^T b \\
&= W(P^T W)^{-1} (M^T)^{-1} W^T A^T b.
\end{aligned}$$

We complete the proof by showing that

$$q = (M^T)^{-1} W^T A^T b,$$

or equivalently

$$M^T q = W^T A^T b.$$

By Lemma 10, $M^T q$ has all entries zero except its first which has the value $\sigma_0 q_0 = \langle A^T b, w_0 \rangle = \| A^T b \|^2$. The entries of $W^T A^T b$ are $\langle w_i, A^T b \rangle$ which are zero except the first which has the value $\| A^T b \|^2$. This proves the following theorem:

Theorem 5. The vector $x = W(P^T W)^{-1} q$ is the solution to the problem

$$\min_{x \in V_k} \| Ax - b \|^2.$$

8 Summary

The final PLS algorithm, expressed in the notation used in this report is given at the beginning of Section 7. As we proved in Section 7, in the end, PLS finds the same solution as the *Orthomin* algorithm. There are two significant caveats to this conclusion however. The first is that as the vectors have been orthogonalized in the usual inner product, and not the required inner product, and the second is that a matrix inverse is required at the final step. This inverse is computationally inefficient and unstable, possibly producing undesirable results. The second point is that if the solution is taken as the linear combination of loading weight vectors, as proposed by some authors, then the solution resembles Craigs method except without the optimality. In the first case, applying the Orthomin algorithm directly would be considerably more efficient, eliminating a fair amount of unnecessary work. In the second case, the PLS solution is a non-optimal iterative method for a problem for which provably optimal solutions exist. Although various interpretations and justifications for the PLS approach have been given, it is our conclusion that the method should be replaced by either of the methods discussed herein.

Appendix A - Implementation

In this section we discuss implementation issues and how we can use the facts presented in previous sections to accelerate PLS as computed by the *Orthomin* algorithm.

Assume first that files are stored on disk in transpose order. This is important since we only use some of the variables in a record and each time we skip a variable when reading from the disk we incur a penalty. It is important to note that the implementation of this algorithm assumes that the data which will be used on any given run fits completely in memory so that indicated matrix multiplies do not come from the disk. In the case this is not possible, a different algorithm will be employed. Discussion of this latter algorithm is postponed temporarily. Recall the *Orthomin* algorithm as presented in Section 5. To implement this algorithm we need to perform the following:

1. Read the chosen variables directly into memory. Read the right hand side vector into memory; this variable is the same length as a single data variable.
2. Perform any necessary transformations to change each variable into floating point.
3. Copy the data to a new location in transposed order. This may be required to avoid cache misses.
4. Call the two copies of the data, given with its transpose, A and A^T . The matrix A has the data points as rows and has n rows and m columns. The right hand vector is called b .
5. We shall require a matrix-vector multiply routine. Call this routine $Matmult(n, m, A, x, b)$ which returns $b = Ax$ when input A and x . This can be built from BLAS routines in Linpack which we comment on below.

6. We require storage for 3 vectors of length m , $x, p, A^T r$, and 2 vectors of length n , which are r, Ap . Storage is also required for 3 inner products which are given by $\langle Ap_i, Ap_i \rangle, \langle Ar_{i-1}, Ar_{i-1} \rangle, \langle Ar_i, Ar_i \rangle$.

The main part of the algorithm with detailed explanations is given below:

Algorithm *Orthomin*

Initially, set

$$\begin{aligned} x_0 &= 0, \\ r_0 &= b, \\ p_0 &= A^T r_0 = A^T b. \end{aligned}$$

Iterate for $1 \leq i \leq m$,

1. $x_{i+1} = x_i + \beta_i p_i$,
2. $r_{i+1} = r_i - \beta_i A p_i$,
3. $p_{i+1} = A^T r_{i+1} + \alpha_{i+1} p_i$,

where $\beta_i = \langle A^T r_i, A^T r_i \rangle / \langle A p_i, A p_i \rangle$ and $\alpha_i = \langle A^T r_i, A^T r_i \rangle / \langle A^T R_{i-1}, A^T r_{i-1} \rangle$.

The *Orthomin* calculations may be implemented using Linpack and the previously mentioned *Matmult* routine. Specific routines of interest are *sdot* to perform inner products, *Matmult* to form $A p$ and $A^T r$, and *saxpy* to form vector add and multiplies.

We now proceed to discuss the operation counts. The major operation counts are determined on each iteration by 2 matrix multiplies, 4 inner products, and 3 applications of *saxpy*. Each iteration costs about $4nm$ floating point operations. It is difficult to determine apriori which stopping criteria to use, and this should probably depend on the application. However, if $A^T A$ is non-singular, then the inner product $\langle A^T r, A^T r \rangle$ should converge to zero and its size is a good indication of the accuracy of the solution. Another approach is to check the magnitude of $x_{i+1} - x_i$ as a jackknife test. At any rate, the total number of calculations is of order $O(4anm)$ where a is the total number of iterations.

One might be tempted to form $A^T A$ in an effort to avoid the $4nm$ work on each iteration. If the number iterations is small, however, this is not a good idea. Forming $A^T A$ takes on order of $(nm)^2$ operations, counting both adds and multiplies, since we need only form half of the symmetric matrix. On the other hand time will be economized if $nm > 4a$ which would hold only for small data sets.

Finally we mention some results on the convergence properties of conjugate gradient. Because each iterate minimizes a polynomial in a matrix C times the original residual, a well-known trick with Chebychev polynomials can be used to bound the residual at each step. Let K^2 be the condition number of the non-singular part of C . That is, is the product of the largest singular value of C times the smallest non-zero singular value. Let y be the vector which minimizes $\|b - Cy\|$ and let $z = Cy - b$. Then $r_n = z - Cy_n$ is the relevant residual in this case. The bound is then given by

$$\|r_n\| < \|r_0\| / \cosh \left(n \log \frac{K-1}{K+1} \right).$$

A more convenient approximation to this last bound can be made when $K \gg 1$. In this case, the number of iterations required to reduce the residual by a factor f is $0.5K \log(2/f)$.

Appendix B - PLS in the case of a binary dependent variable

In the body of the paper it was shown that PLS is a conjugate gradient method of solving the normal equations $A^T A x = A^T b$. In this Appendix, the case where the dependent variable b is binary valued is considered. Let b be an n -dimensional vector whose components take on the two values ϵ_1 and ϵ_2 . Then to satisfy the normalization of the first step of the PLS algorithm, one must rescale b to have zero mean $n_1 \epsilon_1 + n_2 \epsilon_2 = 0$, where n_i is the number of occurrences of ϵ_i . Arbitrary, we choose $\epsilon_1 = 1$, which implies $\epsilon_2 = -n_1/n_2$.

In this case the right hand side of the normal equations reduces to

$$A^T b = \sum_{j=1}^{n_1} A^{(kj)} - \frac{n_1}{n_2} \sum_{j=n_1+1}^n A^{(kj)}$$

where $A^{(ij)}$ is an element of A . This can also be written as

$$A^T b = n_1(\mu_1 - \mu_2)$$

where μ_i is the m -dimensional mean of the independent variables whose corresponding dependent variables have value ϵ_i . That is, $A^T b$ is in the same direction as the line connecting the means of the two classes of b .

The final solution vector obtained by PLS is then given by

$$x \propto S_T^{-1}(\mu_1 - \mu_2)$$

where $x_T = A^T A$ is the (total) covariance matrix.

The form of this solution is very similar to that obtained for the two class version of Fishers linear discriminate analysis. In Fishers discriminate analysis the idea is to find a projection from an m -dimensional

space to an one-dimensional space which leads to the maximum separation between the two classes. The solution [2] to this problem is given by

$$x \propto S_W^{-1}(\mu_1 - \mu_2)$$

where S_W is the within class scatter matrix

$$S_W = S_T - n_1\mu_1\mu_1^T - n_2\mu_2\mu_2^T$$

the second term is called the between class scatter matrix, which measures the separation between the classes. What Fishers linear discriminate analysis does is to maximize the ratio of between class scatter to within class scatter. Notice that in the case when the between class scatter is small the final solution vector found by the PLS algorithm is an approximation to the direction found by Fishers linear discriminate analysis.

The first solution vector x_1 found by the PLS algorithm also has an interesting interpretation in the case where b is binary valued. Recall from equation (6.4) that the PLS projection equation is given by

$$\tilde{x} = W(P^TW)^{-1}q$$

for the first iteration W is an $m \times 1$ matrix, therefore it is easy to show that $\tilde{x}$ is in the same direction as A^Tb . Recall that A^Tb is in the same direction as $\mu_1 - \mu_2$, that is, in this case the first solution vector found by PLS is just the line connecting the means. While the final solution is in the $S_T^{-1}(\mu_1 - \mu_2)$ direction. In other words, the algorithm starts with the line connecting the means as an initial guess to the solution, but converges to the least squares solution.

References

1. O. Axelsson, *Iterative Solution Methods*, Cambridge University Press, Cambridge 1996.
2. R. Duda, P. Hart, *Pattern Classification and Scene Analysis*, John Wiley and Sons, New York, 1973.
3. W. Dillon, M. Goldstein, *Multivariate Analysis: Methods and Applications*, Wiley Series in Probability and Mathematical Statistics, John Wiley and Sons, New York, 1984.
4. V. Faber, T. Manteuffel, Necessary and Sufficient Conditions for the Existence of a Conjugate Gradient Method, *SIAM J. Numer. Anal.*, Vol 21, No.2, April 1984.
5. V. Faber, T. Manteuffel, Orthogonal Error Methods, *SIAM J. Numer. Anal.*, Vol 24, No. 1, February 1987.
6. P. Garthwaite, An Interpretation of Partial Least Squares, *Journal of the American Statistical Association*, Vol. 89, No. 425, March 1994.
7. G. Golub, C. Van Loan, *Matrix Computations*, The Johns Hopkins University Press, 1989
8. I. Helland, On the Structure of Partial Least Squares Regression, *Commun. Statist. Simula.* 17(2), pp. 581-607 (1988).
9. H. Martens, T. Naes, *Multivariate Calibration*, John Wiley and Sons, New York, 1989.