

Math 642:550 — Summer 2006
MTTh 6:00–8:30 PM Hill 705
Prof. Bumby

Supplement 8, The Pseudoinverse

1. Introduction

The text uses the **Singular Value Decomposition** to describe the **pseudoinverse**, since it gives an easy proof of existence, but it is not necessary for computation. We describe how to compute values of the pseudoinverse using only rational operations. This makes this operation accessible to hand computation in many problems. However, machine computation is more concerned with controlling error. Here, the use of the SVD may be preferred.

2. Least squares solutions

A special case of the pseudoinverse is the **Least Squares Solution** of an **overdetermined** system. Such systems typically arise when trying to get the **best fit** to data using some special type of function. For example, **linear regression** aims to draw a line $y = ax + b$ through a family of points (x_i, y_i) so that the **set of values** $\{y_i - (ax_i + b)\}$ is small in an appropriate sense. Gauss proposed finding a and b so that

$$\sum_i (y_i - (ax_i + b))^2 \quad (*)$$

is minimized. This has the advantage that these values are easily found. The name **least squares** comes from this aim to minimize a sum of squares.

It is common in calculus courses to obtain a **formula** for the desired values of a and b . However, this is a **linear algebra** course, so we shall relate this problem to linear transformations and the geometry of vector spaces.

Thus, we make the desired quantities (a, b) into the components of a vector in \mathbb{R}^2 , and the values $\langle ax_i + b \rangle$ and $\langle y_i \rangle$ determine vectors in \mathbb{R}^n , with n equal to the number of data points. The x_i then determine a linear transformation from \mathbb{R}^2 to \mathbb{R}^n , and $\langle ax_i + b \rangle$ is an element of the **column space** of the matrix M with rows $[1, x_i]$ that represents the linear transformation.

If $\langle y_i \rangle$ were in this column space, then we could make all $y_i - (ax_i + b)$ equal to zero, but this is unlikely to happen if n is large. Instead, we note the equation $(*)$ is a formula for the square of the distance between $\langle y_i \rangle$ and a point of the column space. The Pythagorean theorem tells us that distance is minimized by the point in the column space on a line perpendicular to the column space. Such lines belong to the **left nullspace** of the matrix M , which may also be described as the **nullspace of the transpose** of M . To solve a least squares problem, $\langle y_i \rangle$ should be projected into the column space of M and that projection expressed in terms of the column space of M . For our linear regression problems, the columns of M will be linearly independent as long as the x_i take at least two different values. Then, there will be a unique solution (a, b) expressing the projection into the column space as a linear combination of the columns. The same analysis applies to any matrix M with **linearly independent columns**.

3. The normal equations

It is a **useful fiction** to pretend that we are trying to solve a system of the form $Mx = c$, and to multiply this on the left by M^T to get $M^T Mx = M^T c$. If M has linearly independent columns, then $M^T M$ is nonsingular, since a vector x in its nullspace satisfies $M^T Mx = \mathbf{0}$, which implies that $(Mx)^T(Mx) = x^T(M^T Mx) = 0$. However, this says that the length of Mx is zero, which implies that x itself is the zero vector. This fiction is only useful if we pick M^T as the left multiplier, so we need to make that part of the solution process.

The system $M^T Mx = M^T c$, called the **normal equations** associated with M and c , has a unique solution $x^{(LS)}$. The normal equations say that $M^T(Mx^{(LS)} - c) = 0$, which says that $Mx^{(LS)} - c$ is orthogonal to the column space of M . In other words, $Mx^{(LS)}$ is equal to the desired projection of c into the column space of M .

For the **toy problems** met in homework exercises that can be solved exactly, this is usually the best way to find a least-squares solution.

4. Example

Let

$$M = \begin{bmatrix} 5 & -2 \\ -1 & -6 \\ -3 & -8 \\ 4 & -1 \\ 7 & 2 \end{bmatrix}, \quad c = \begin{bmatrix} -45 \\ -49 \\ -77 \\ -9 \\ 33 \end{bmatrix}$$

Then,

$$M^T M = \begin{bmatrix} 100 & 30 \\ 30 & 109 \end{bmatrix}, \quad M^T c = \begin{bmatrix} 250 \\ 1075 \end{bmatrix}$$

so that the normal equations are

$$\begin{bmatrix} 100 & 30 \\ 30 & 109 \end{bmatrix} x = \begin{bmatrix} 250 \\ 1075 \end{bmatrix}$$

whose solution is

$$x = \begin{bmatrix} -1/2 \\ 10 \end{bmatrix}$$

This solution is checked by finding the **residual**

$$c - Mx = \frac{1}{2} \begin{bmatrix} -45 \\ 21 \\ 3 \\ 6 \\ 33 \end{bmatrix}.$$

This isn't particularly small — it doesn't have to be — but it is perpendicular to the column space of M , so it confirms that x is the “least squares solution” of $Mx = c$.

5. Numerical considerations

One thing that should be noticed is that the numbers appearing in the normal equations $M^T Mx = M^T c$ are roughly the **squares** of the numbers appearing in M and c . This was not a major concern in our example, since the numbers were rational and could be written exactly with little effort.

However, if the matrices are only known approximately, the behavior of the error depends on the actual steps in the computation. The discussion of **condition number** in section 7.2 of the text gives a rough idea of how to predict when there will be a significant magnification of error in a computation. In particular, the application to the “least-squares” problem is discussed on page 356. Note 2 on that page says that, because the **condition number** is the ratio of the largest eigenvalue to the smallest one, squaring the matrix will square this ratio, and this may seriously magnify the errors. A more detailed discussion can be found in section 5.3 of Gene H. Golub & Charles F. Van Loan, *Matrix Computations*, Johns Hopkins, 1989 (ISBN 0-8018-3772-3 (hardcover) and 0-8018-3739-1 (paperback)). After careful analysis of two methods of solving the “least squares” problem, those authors conclude: ‘At the very minimum, this discussion should convince you how difficult it can be to choose the “right” algorithm!’. More details will be given in our discussion of chapter 7 of the textbook.

6. The general case

A similar method applies when the columns of M are not linearly independent. Again, the first step is to project c into the column space. This can be done by forming the system $M^T Mx = M^T c$, but this system now has a **singular matrix of coefficients**. Thus, the solutions will fill a **translate of the nullspace** of $M^T M$. Again, $M^T Mx = \mathbf{0}$ implies $x^T M^T Mx = 0$, which implies that Mx is the zero vector. Hence, the solutions fill a translate of the nullspace of M . To select a **canonical** (or **special**) element of this set, we choose the vector x **of shortest length**. The usual analysis of extrema of distance reveals that this is the vector in this set that is perpendicular to the nullspace of M . Our knowledge of the **four fundamental subspaces** tells us that the space perpendicular to the nullspace is the **row space** of M . To get our special solution, we need to find one solution and project it into the row space of M . Any solution that we find will have the same projection. This analysis shows that the solution of such generalized least-squares problems are unique.

Moreover, the solution of this least-squares problem may be described as the composition of linear transformations. First, project into the column space of M ; then, apply the **inverse** of a mapping from the r dimensional row space of M to its r dimensional column space; finally, express the resulting vector in terms of the original basis of \mathbb{R}^n . An n by m matrix expressing this linear transformation in terms of the given bases of \mathbb{R}^n and \mathbb{R}^m is called the **pseudoinverse** of M .

7. Computing solutions

The QR factorization of an m by n matrix M of rank r , which can be found either by the **Gram-Schmidt** process or by building an orthogonal matrix Q from **Householder matrices**, produces an orthogonal basis for the column space of a given matrix M . The Gram-Schmidt process stops at this point, but the Householder matrices lead to an orthonormal basis for all of \mathbb{R}^m whose first r columns are a basis for the column space, with the remaining $m - r$ vectors being a basis for its **orthogonal complement**, the **left nullspace**. Multiplying any given vector v by Q^T gives the components of v with respect to this basis, allowing the projection into the column space to be found.

The Gram-Schmidt process is better for hand computation since it only works with rational numbers, except for square roots needed to normalize the vectors in the orthogonal basis if M has rational entries.

These normalization factors are only a theoretical convenience: they disappear in the expression for the projection. On the other hand, we have seen that Householder matrices sometimes make essential use of irrational quantities. Their effect will also disappear eventually, but the **exact** description of the orthonormal basis used to compute the projection is not guaranteed to be simple.

If one has the QR factorization of M with an m by r factor Q and an r by n factor R , the vectors x mapping to the projection of c into the column space of M are the solutions of $Rx = Q^T c$. In order to find our preferred solution, we need to find a solution in the **row space** of M , which is also the row space of R . Since the rows of R are linearly independent, we can use them as a basis for the row space. This suggests writing $x = R^T z$, so that our equation becomes $RR^T z = Q^T c$. If only the r nonzero rows of R are used, RR^T is nonsingular and this system has a unique solution. That solution is z — the components of x with respect to our chosen basis for the row space on M — so we need to multiply by R^T to obtain x .

This process requires that we solve a system whose coefficient matrix is RR^T , which is no problem when we are solving equations exactly. However, as with the **normal equations**, there may be difficulty when solving a large system approximately. If this is a concern, the solution should find any solution of $Rx = Q^T c$ and project it from \mathbb{R}^n into the **row space** of M , which is also the row space of R . This can be done by finding a QR factorization of R^T . Putting these two factorizations together gives what Golub and Van Loan call a **Complete orthogonal decomposition**. Here, $M = Q_1 T Q_2^T$, where T is **lower triangular** (since it is the transpose of the R factor in the QR factorization of the transpose of the R factor in the QR factorization of M).

There are many different complete orthogonal decompositions of M , but they all lead to the same pseudoinverse the underlying linear transformation is uniquely defined. The **singular value decomposition** is an example of a complete orthogonal decomposition. The construction in the textbook is an example of the general construction that we have just described. When dealing with approximate quantities using a computer, a routine for computing an approximate **SVD** is likely to be available, so the calculation of the pseudoinverse, or even the solution of a single general least-squares problem, can be found in terms of the SVD. However, the exact solution of the **toy problems** used as exercises or exam problems in courses can be found by using the Gram-Schmidt process to compute the projections.

Other approaches are possible. If one forms an orthogonal basis starting from the rows of R **in reverse order**, a factorization similar to the complete orthogonal decomposition, but with an **upper** triangular matrix as middle factor, will be found. The exact form of this middle factor is not important; the only requirement is that it be easy to solve a system of equations with this matrix of coefficients.

End of Supplement