1 Application of Linear Algebra in Infants' Autism Detection

There is an ongoing autism research led by Dr.Guillermo Sapiro, a professor in Pratt School of Engineering at Duke University. Using image processing, he attempts to program a computer to detect whether babies (around eight to fourteen month of age) display a sign of autism. This very early detection enables doctors to train these babies (when their brain plasticity is high) to behave in ways to counter the behavioral limitations autism imposes, thus allowing these babies to act more normally as they grow up. One of the behavioral cues, which Dr.Sapiro looks for, is the blinking of the babies' eyes. This portion of the lecture notes focuses on the creation of an image processing algorithm for the detection of the blinking, in particular for the determination of whether an eye is open or closed, by applying concepts of linear algebra.

The input to the algorithm is a picture of an eye. Each input image consists of 30(width) x30(height)= 900 pixels. Each pixel carries a numerical value in the range of 0 to 255, which represents the combination light intensity detected by the red, blue, and green light receptor channels of a camera. The significance of each pixel varies. The image of an open eye will have two major dark edges around the eyes with a large pupil in the middle. On the other hand, the image of a closed eye will have only one dark edge with no pupil. There is a pattern associated with each of these two categories of images.

The challenge for the algorithm is to determine the pattern from the pixel data with a high probability of success. In our model, this is achieved with the aid of properly chosen weights for the individual pixels. The determination between an open or closed eye is then made in the following simple way. The algorithm multiplies the numerical pixel value by the pixel weight and then sums these products over all pixels. If the result of the summation is positive, the output of the algorithm is the integer '1' and the eye is determined to be open. If the sum is negative, the output of the algorithm is the integer '-1' and the eye is determined to be closed.

In order to describe and justify the selection of the weights, we represent the weights as the components of a vector x (1) . The pixel sequencing corresponding to the sequencing of the vector entries is immaterial. We may start with the weights of the pixels of the first row of the image, continue with the weights of the second row, then the third row, etc. We arrange the 900 pixel values into another vector, which we call a, in the same pixel sequencing (2). We consider x as column vector and a as a row vector. We can now express the sum described above as the matrix product ax. This is simply the dot product of the two vectors. Denoting the matrix product (a scalar) with the letter b, we obtain the equation

$$ax = b$$

In order to "train" our model to be an accurate predictor of the state of the eye, in other words, in order to determine an appropriate weight vector x, we utilize a large number of images of open and closed eyes. We also need a large number of images in order to test the accuracy of the trained model. The eye data set used for this project consists of 10,000 eye

images that are either fully closed or fully open. Of the 10,000 images, 7,000 are used to train the model to learn which eyes are open and closed, and the remaining 3,000 images are used for testing how well the trained model performs (generalizes) on a new data set. The terms "training" and "testing" are widely used in data analysis fields, especially in machine learning. The nature of the training is described below.

Each image satisfies the above equation with its own vector a and with the right side of the equation, b, set equal to the "ideal" value, that is '1' if the image depicts an open eye, and '-1' if the image depicts a closed eye. Thus, we have 7,000 equations (one per image) for the the 900 unknown weight entries (one per pixel) of the vector x. We write these equations as a single matrix equation, b_m.

$$Mx = b \tag{1}$$

in which M is the 7,000x900 matrix that has the 7,000 row vectors ('a') for its rows, x is the 900x1 column vector for the weights of 900 pixels, and b is the 7000-column vector of the corresponding values of b (i.e. 1 or -1) that represents the state (open/closed) of the 7000 training image. So, each row of the matrix M represents an image, while each column represents a specific pixel within each image.

The matrix M has many more rows than columns; in terms of scalar equations, the equations far outnumber the unknowns. Thus, for all but unrealistic contrived data, the system of equations is inconsistent, the "ideal" vector b lies outside the range of the matrix M and the matrix equation has no solution. In an introductory linear algebra classes, such overdetermined system may seem useless since it does not yield a unique solution. However, in data analysis either in research or industrial application, an overdetermined system is actually very useful because it allows training massive amount of data. In general, the more training data there are, the more stable the results is. This blinking project uses 7,000 images for training, which is enough data to have a computer detect blinking with about 90 percent accuracy when tested with the remaining 3,000 images that are not used for training.

The optimization strategy of the model in order to overcome the overdeterminacy of the system is to replace the vector b with the vector in the range of M that is closest to b (has the shortest Euclidean distance from b); all this within the vector space $\mathbb{R}^{7000}$ of all real vectors with 7,000 entries, of which the range of M is a subspace. We denote this vector by, b_m. Clearly b_m is the orthogonal projection of b on the range of the matrix M. In other words, b_m is identified as the unique point, at which the line drawn from the point b perpendicularly to the range of M intersects the range of M.

We have posed two conditions: (1) $Mx = b_m$ and (2) the vector $b - b_m$ is orthogonal to the range of M. We recall, that the orthogonal complement of the range of M is the null space of the transpose M^T of M, thus, the second condition is equivalent to $b - b_m$ belonging to the null space of the matrix M^T. Our two conditions take the form

$$Mx = b_m \text{ and } M^T(b - b_m) = 0 \tag{2}$$

We can eliminate the vector b_m from the two equations by applying the matrix M^T to (1). Then, we obtain

$$M^T M x = M^T b \tag{3}$$

This is precisely the equation obtained when the classic method of Least Squares is used to solve the above eye-blinking autism problem. Our approach gives a geometric interpretation of the Least Squares method.

The required vector of weights x is obtained by solving this linear system. $M^T M$ is a real symmetric matrix. If it is invertible, we can obtain the vector of weights x by formula,

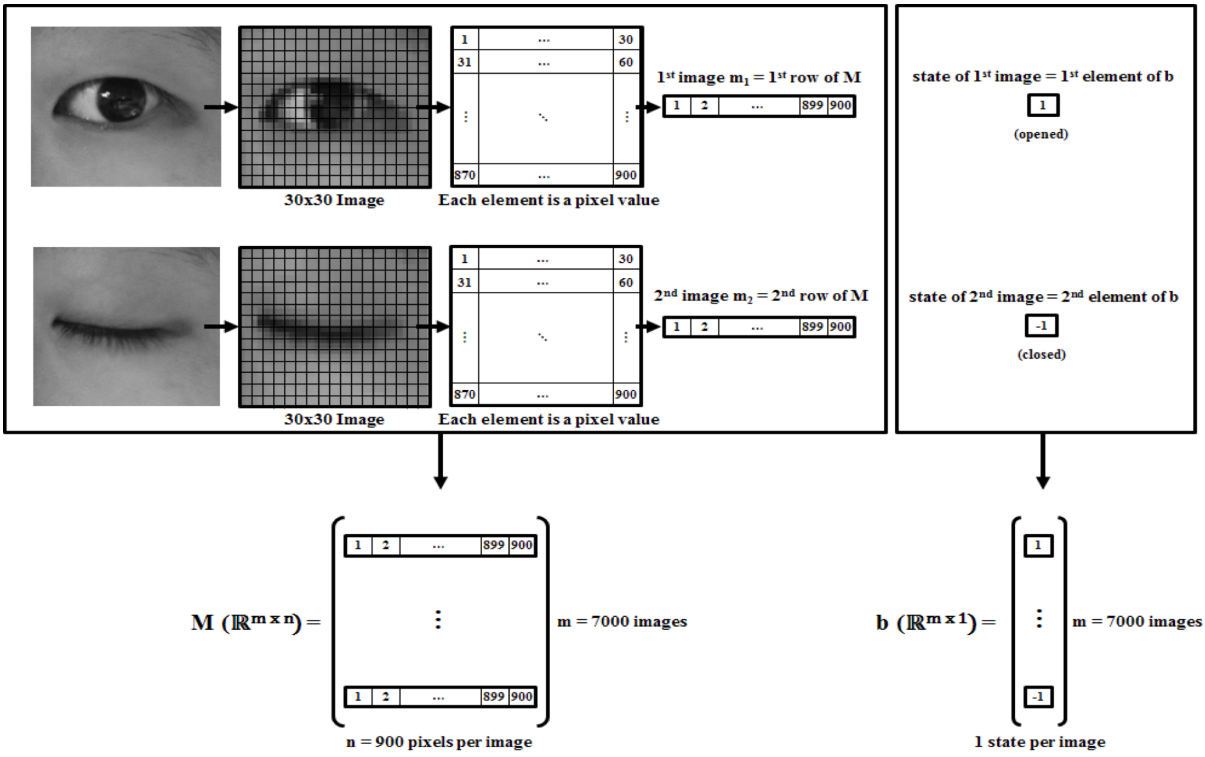
$$x = (M^T M)^{-1} M^T b \tag{4}$$

If the matrix $M^T M$ is non-invertible, we obtain a unique solution x by making the additional requirement that x be orthogonal to the null space of M (a subspace of $\mathbb{R}^{900}$) in the vector space $\mathbb{R}^{900}$. This places x into the the range of M^T, due to the fact that this subspace is the orthogonal complement of the null space of M in $\mathbb{R}^{900}$. This choice also guarantees that the x chosen in this way has the least magnitude of all the solutions to (3).

Expert suggestion in scientific computing: In practice, matrix $M^T M$ is often poorly conditioned (nearly singular) and can lead to significant errors in the numerical solution of (3),due to rounding error. While these errors may be reduced using pivoting in combination with Gauss Elimination, it is generally better to solve (3) using the Householder transformation, as this produces less rounding error, or better still by Singular Value Decomposition which will highlight any redundant or nearly redundant variables in x.

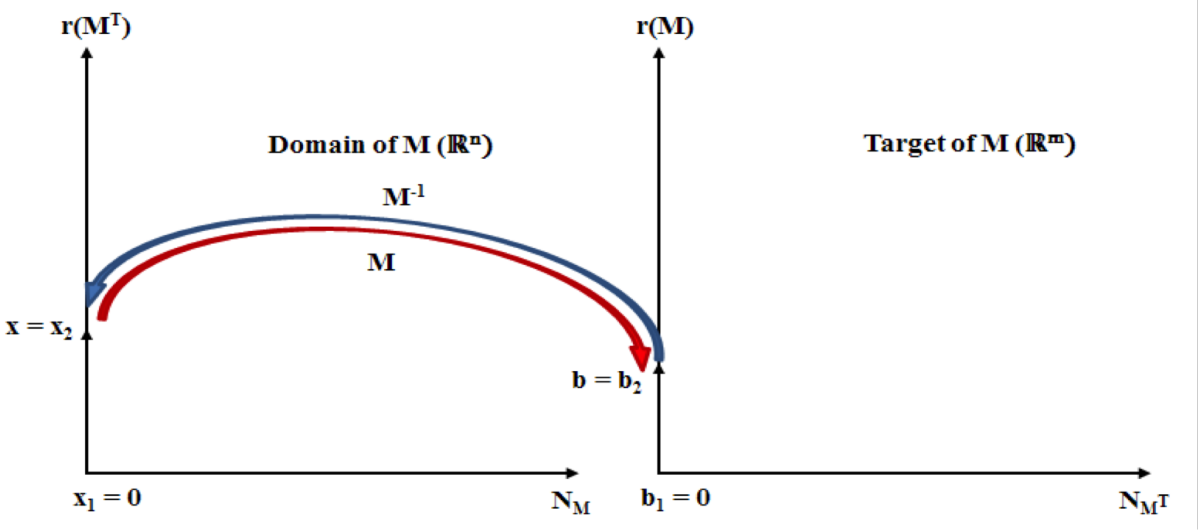
2 Methods

In order to efficiently train all these 7,000 images, we recreated a matrix, M,(instead of matrix A) that holds all these 7,000 images (refer to the example diagram below). Each 30 x 30 image matrix is reshaped into 1 x 900 vector and then was attached to matrix M. Now that we have a system of equation, $Mx = b$, x represents a weight vector of size 900 x 1. And, the output vector, b, represents a vector of size 7,000 x 1 with values of either -1 or 1. As a result, due to immense amount of training data to compute the weight vector, x, we now have to consider a heavily overdetermined system in which there are lot more rows than columns. There is no inverse for this matrix M. To recapitulate, now we are faced with one linear algebra question: $Mx = b$. Our ultimate objective is to find the weight vector x. However, the matrix M is non-invertible due to its asymmetric size (as shown in the diagram below).



Now,our algorithm is no longer a computer science problem. It is simply a linear algebra problem with a 30x30 matrix. Do you remember the simple system of equation: Ax = b? This will be the primary concept! The matrix, A, in this equation is the eye image matrix. The vector, x, represents the "weight" of each pixel. Each pixel would have a different significance in determining the state of the eye Lastly, the vector, b, represents the state of the image with 1 equivalent to "open" and 1, to "closed." So, multiplying the inverse of A to both sides of the equation will get us the desired weight vector, x. Once x is acquired, making decision about an input eye image is easy. By multiplying the input image matrix with the weight vector, if the newly computed output vector, b, is closer to 1, the image has an open eye. If closer to 1, then the image is a closed eye. However, there is a problem. We need more restrictions on M to find x. Now, the objective is to compute vector x. Suppose M is an invertible square matrix (thus full rank), computing x would be much simpler: $x = M^{-1}b$.

In terms of the four basic subspace, this operation can be represented as the following.

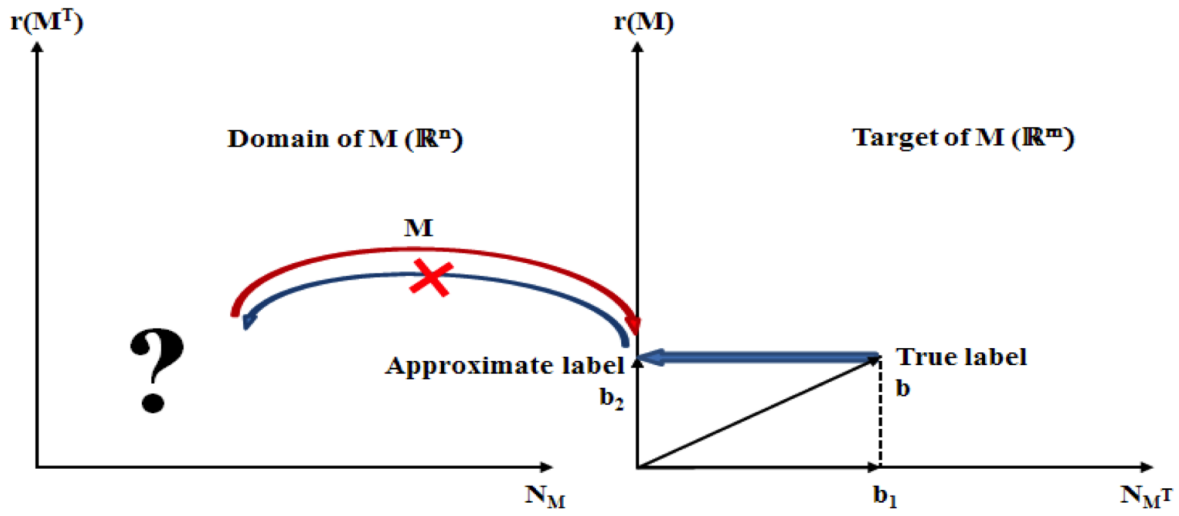


However, in data analysis either in research or industry, it is almost never the case that the amount of data (in our case, 7,000 eye images) matches the dimensionality of the data (900 pixels per image). It is either that the amount data is greater than the dimensionality, which corresponds to overdetermined systems, or the amount of data is smaller than the dimensionality, which correspond to underdetermined systems.

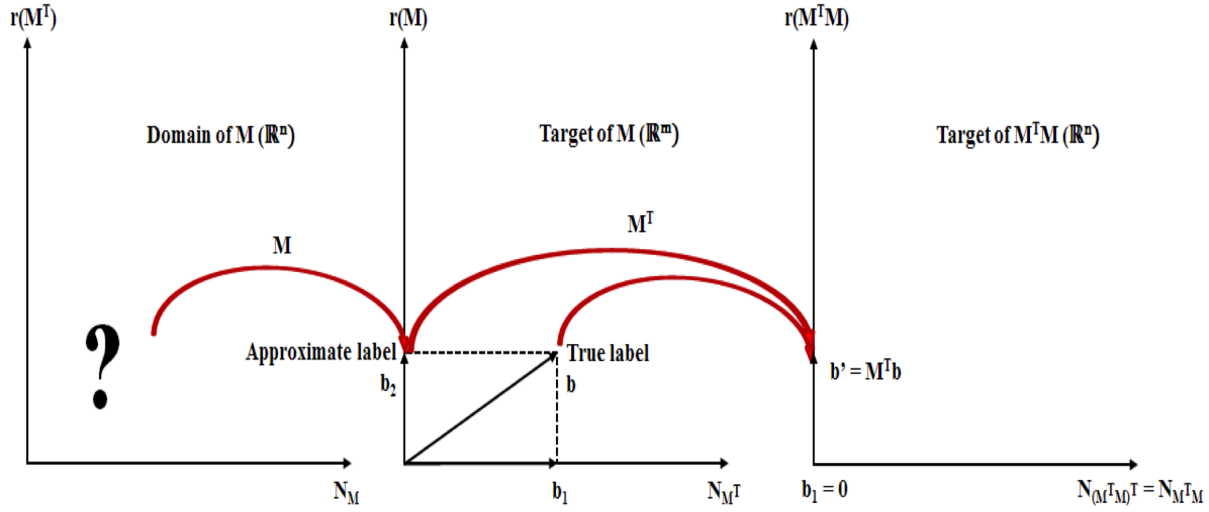
In an introductory linear algebra classes, such overdetermined system may seem useless since it does not yield a unique solution. However, in data analysis either in research or industry, overdetermined system is actually very useful because it allows training massive amount of data. In general, the more training data there are, the more stable results is. Out of 10,000 available eye images, this blinking project uses 7,000 images for training, which is enough data to have a computer detect blinking with about 90 percent accuracy (when tested with the remaining 3,000 images that are not used for training).

We will derive the solution to an overdetermined system using the four fundamental subspaces.

1) In this problem, there are 7000 equations and 900 unknowns (overdetermined), and the system does not have an exact solution. To get a unique solution, by definition, b has to be in the range of M. Since an exact solution cannot be acquired, certain approximation has to be made. One of the best approximation would be to use a projected b onto the range of M, which is closest to b and is on the range of M. (How do we explain that this is the best approximation without mentioning Least Square Method?) However, even if we use an approximate b that is in the range of M, we cannot compute x directly from the inverse of M because M is not a square matrix and it is non-invertible.



2. This problem can be reformulated by multiplying M^T on both sides of the equation such that $M^T Mx = M^T b$. The effects of applying M^T to both side of the linear equation are as follows:



i) M^T maps all values of $b = cb_1 + b_2$ in $\mathbb{R}^m$ for any c to a single value $b' = M^T b_2$ in $\mathbb{R}^n$ because $M^T b = M^T(cb_1 + b_2) = cM^T b_1 + M^T b_2 = M^T b_2 = b'$. Therefore, multiplying M^T to both b and b_2 maps these to the same point b‘ that is in the range of $M^T M$.

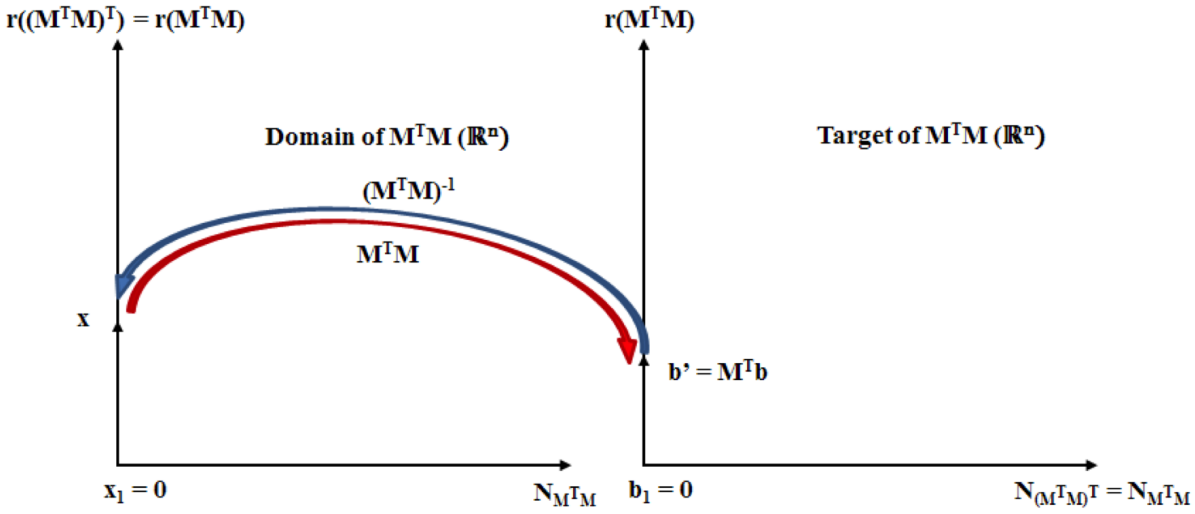
ii) This b’ is in the range of $M^T M$. $(M^T Mx = b')$

iii) Matrix $M^T M$ is a square, symmetric matrix since transpose of this matrix is itself. We

can also safely assume the columns of matrix M are linearly independent. This assumption is reasonable because eye images are all distinct, meaning it is highly unlikely that pixel related values of an image is linearly dependent on other image's values. As a result, we assume that M^TM is invertible.

iv) Since matrix M^TM is invertible, the null space of M^TM (and that of its transpose $(M^TM)^T = M^TM$) is trivial.

3. From the properties above, we can redraw the four basic subspaces of the equation $M^Tx = M^Tb$.



4. Using this method, we can compute x using the equation, $x = (M^TM)^{-1}M^Tb$ This method approximates b as b_2 as shown above. Thus, the value of x that approximates the state (namely, b) of given set of eye images (namely, M) can be computed. Now that we have computed the x from the 7,000 training data, it is used on the remaining 3,000 testing data to see how well x can approximate the true states of the testing data.

$$M_{test}x = M_2(M^TM)^{-1}M^Tb = b_{test}$$

where M_{test} is the 3,000 x 900 matrix of testing data, x is the computed 900 x 1 weight vector and b_{test} is the 3,000 x 1 response vector. It would be ideal if b_{test} yields only 1 or -1 values. However, b_{test} is the best approximated value, which consist any real number. There has to be a criterion for a binary classification of open and closed state. Here, if an element in b_{test} is greater than 0, then the corresponding image is classified as open and denoted as '1' while if smaller than 0, it is classified as closed eye '-1'.

The closer b_test is to the true states of the 3,000 testing images, the more accurate the prediction would be. In addition, the matrix $(M^T M)^{-1} M^T$ is also called the Moore-Penrose Pseudo Inverse. In data mining and machine learning, the computation of vector x from a given M and b is called "Training" of a classifier where M and b are called "Training Data". Also, predicting the state of a new input data is called "Testing" where a new input M_{test} and its corresponding eye states are called the "Testing Data or Untrained Data."

The result of testing with the remaining 3,000 untrained images is approximately 90 percent.

3 Conclusion

Different methods are used to cope with overdetermined system. A widely used method is to compute the least square solution. That is, even though no exact solution exists for, we try to compute the "best" approximate solution possible. "Best" is this case refers to minimizing the sum of the squared error, (formula for least squares solution) Computing the leasts quare solution is equivalent to computing the pseudo-inverse of the matrix M in matrix terminology. $x = M^{**}b$ where $M^{**} = (M^T M)^{-1} M^T$.