CS 542G: More QR, High Dimensional Data, PCA, SVD

Robert Bridson

October 8, 2008

1 More on the QR Factorization

Last time we looked at two useful algorithms for computing the QR factorization: Modified Gram-Schmidt and Householder. They in fact give slightly different versions of the factorization.

If A is $n \times k$ and has rank k, then MGS constructs a $k \times k$ upper triangular matrix R so that the columns of $Q = AR^{-1}$ are orthonormal. In this case, Q is $n \times k$, just like A.

However, the Householder approach produces a sequence of $n \times n$ orthogonal matrices (the Householder reflections) whose product is Q, so that $Q^T A$ is upper triangular.[1] Clearly this Q is square, of dimension $n \times n$, and is fully orthogonal. The R must be $n \times k$—not necessarily square, but with zeros below the diagonal all the way down (it will have $n - k$ all zero rows at the bottom).

Due to the differing sizes of Q, the Householder QR is sometimes called the "full QR", and the MGS version the "economy QR". At any rate, up to signs, the first k columns of the full Q will match the economy Q.

The story is a little different when A does *not* have full column rank. The columns of Q in MGS are linear combinations of the columns of A, and so can't have rank greater than A: in the rank-deficient case, the MGS Q will necessarily have zero columns—and to compute it fully, special checks must be made in the code to avoid trying to normalize zero intermediate vectors $\tilde{q}$.

However, for Householder QR, as long as we similarly skip over reflections for zero vectors (or take the vector v to be a column of the identity matrix instead), all the factors are perfectly orthogonal. This means their product Q is a full rank, invertible, orthogonal matrix—no matter how degenerate A is. This will come in handy later when solving eigenproblems.

[1] But remember, that unless we really need the matrix entries of Q, it's probably best not to multiply out the Householder reflections.

2 High Dimensional Data

Let's now look for an even tougher problem. For example, what if we assume not only are there errors in the measured values $\{f_i\}$ but also in the locations $\{x_i\}$ where they were measured? We can potentially make a better fit to the data if we take these errors into account as well, finding an estimate both of the true function value and the true position of each data point. You could think of generalizing least-squares to mean minimizing the distance from the graph of the estimated function $(x, f(x))$ to the data points (x_i, f_i). This is sometimes termed **Total Least Squares**.

One effective way to approach this, and get into an even larger class of problems, is to simply lump the x values in with the data: treat the input $\{(x_i, f_i)\}$ simply as a collection of points in some higher dimensional space. The goal is then to find a plausible "shape" or manifold that roughly contains the data points. This is of general interest even when there is no obvious functional relationship between some coordinates and others.

For example, imagine studying "motion capture data", i.e. the angles of the joints of a human in a long sequence of different poses (as the human performs some action). Each data point is one pose, a vector of joint angles, but it's not clear that some of those angles should be thought of as functions of the other angles. We do expect there to be correlations, however, and it can be very instructive and useful to figure those correlations out.

This veers towards the whole topic of machine learning, which involves algorithms which try to determine simpler (e.g. lower dimensional) models for complex high dimensional data. We'll only just broach the subject by looking at a particularly simple case: **Principal Components Analysis** (PCA).

3 PCA

The big assumption underlying PCA is that the input data, which lives in $\mathbb{R}^m$, is to a good approximation restricted to a lower k-dimensional subspace; it finds this subspace in a least-squares sense. Only looking at subspaces means we're both assuming the data is "flat" (i.e. not from a curved manifold) and also that this manifold goes through the origin. To stand a better chance of meeting up with the second assumption about the origin, often the first step of PCA is to subtract the mean off all the data points, roughly centering them on the origin.

We can represent a k-dimensional subspace as the span of a set of k basis vectors in $\mathbb{R}^m$. The problem then becomes to approximate every data point (each an m-dimensional vector) as a linear combination of k vectors: determining both the coefficients in the linear combination and the set of basis

vectors is the difficulty.

Let's first tackle the case where $k = 1$, i.e. where we expect our data points to lie in a one dimensional subspace which is just a line in $\mathbf{R}^m$. We need to find an m-dimensional basis vector u for this subspace, and for each of the n data points a coefficient so that $f_j \approx w_j u$. We can combine these coefficients into an n-dimensional vector w, and all the data points as columns in an $m \times n$ matrix A. In other words, we approximate the i'th coordinate in the data and the j'th data point as

$$A_{ij} \approx u_i w_j$$

and in general are approximating the entire matrix A with the outer product of the two vectors:

$$A \approx uw^T$$

We can also phrase this as approximating A with a rank one matrix.

We again have to decide what makes a good approximation, with a least-squares type approach being the simplest mathematically again. We turn therefore to the Frobenius norm, giving the following least-squares problem:

$$\begin{aligned} & \min_{u,w} \|A - uw^T\|_F^2 \\ \Leftrightarrow \quad & \min_{u,w} \sum_{i=1}^{m} \sum_{j=1}^{n} (A_{ij} - u_i w_j)^2 \end{aligned}$$

This is a bit different than our previous least-squares problems, since it's not linear: the variables are u and w, and they appear as a product in the residual. Nonetheless, we will get to the point soon of figuring out an elegant answer—but it will involve more than just solving a linear system.

This isn't too difficult, in fact, to tackle directly: a good exercise is to work out what the optimal w would be assuming the optimal u is already known, which does have a simple answer ($w = A^T u / \|u\|_2^2$). Then plug this formula for w in, and try to optimize with respect to u...

Getting back to the original problem instead, if we are looking for a k-dimensional subspace to approximate the data (k greater than 1), we can take the same approach. Let U be an $m \times k$ matrix, with its columns being the basis of the k-dimensional subspace; approximate the data points with linear combinations of those basis vectors, and assemble the resulting coefficients in a $k \times n$ matrix W. Then the least-squares problem is:

$$\begin{aligned} & \min_{U,W} \sum_{i=1}^{m} \sum_{j=1}^{n} \left(A_{ij} - \sum_{s=1}^{k} U_{is} W_{sj} \right)^2 \\ \Leftrightarrow \quad & \min_{U,W} \|A - UW^T\|_F^2 \end{aligned}$$

A more abstract way of phrasing this is approximating A with a rank k matrix.

We'll now take a detour to look at a very important matrix factorization which happens to solve this problem amongst many others.

4 The Singular Value Decomposition

The **Singular Value Decomposition** (SVD) of an $m \times n$ matrix A is usually defined as the product

$$A = U\Sigma V^T$$

where U is an orthogonal $m \times m$ matrix, V is an orthogonal $n \times n$ matrix, and Σ is a diagonal $m \times n$ matrix. The columns of U and V are called the (left and right) singular vectors; the diagonal entries $(\sigma_1, \sigma_2, \ldots)$ of Σ are called the **singular values** of A. By convention, the singular values should all be non-negative and in decreasing order:

$$\sigma_1 \geq \sigma_2 \geq \ldots \geq 0$$

The SVD exists for any matrix.

We can prove the existence of the SVD with induction. The base case, where either m or n is 1 (i.e. A is a row or column vector), is fairly trivial: one of U or V is just the scalar number 1, the sole singualr value is the 2-norm of the vector, etc. For larger m and n, first let $\sigma_1 = \|A\|_2$ be the 2-norm of A, and let unit-length vectors u_1 and v_1 satisfy $\sigma_1 u_1 = Av_1$ (i.e. v_1 is a unit-length vector which maximizes $\|Av_1\|_2$). Using Householder QR, for example, we can create orthogonal matrices $\tilde{U}$ and $\tilde{V}$ with u_1 and v_1 as the first columns. Then take a look at $\tilde{U}^T A \tilde{V}$:

$$\left(\begin{array}{c|c} u_1 & \bar{U} \end{array}\right)^T A \left(\begin{array}{c|c} v_1 & \bar{V} \end{array}\right) = \left(\begin{array}{c|c} \sigma_1 & w \\ \hline 0 & \bar{A} \end{array}\right)$$

The first column has to be of this form since $Av_1 = \sigma_1 u_1$, and $\tilde{U}$ is an orthogonal matrix with u_1 as its first column.

The matrix we formed has the same 2-norm as A, which is σ_1, so therefore if we multiply any vector—such as (σ_1, w)—by A then it can't be amplified by more than that:

$$\left\| \left(\begin{array}{c|c} \sigma_1 & w \\ \hline 0 & \bar{A} \end{array}\right) \begin{pmatrix} \sigma_1 \\ w^T \end{pmatrix} \right\|_2^2 \leq \sigma_1^2 \left\| \begin{pmatrix} \sigma_1 \\ w^T \end{pmatrix} \right\|_2^2$$

Expanding this out gives:

$$\left\| \begin{pmatrix} \sigma_1^2 + \|w\|^2 \\ \bar{A}w^T \end{pmatrix} \right\|_2^2 = (\sigma_1^2 + \|w\|^2)^2 + \|\bar{A}w^T\|^2 \leq \sigma_1^2(\sigma_1^2 + \|w\|^2)$$

which reduces to:

$$\sigma_1^4 + 2\sigma_1^2\|w\|^2 + \|w\|^4 + \|\bar{A}w^T\|^2 \leq \sigma_1^4 + \sigma_1^2\|w\|^2$$

The only way this can be satisfied is if $w = 0$. Since $\bar{A}$ is smaller, by induction we can find its SVD, and with the appropriate multiplications produce an orthogonal U and V (with first columns u_1 and v_1) where U^TAV is diagonal. It's trivial to ensure that this diagonal matrix, which we will call Σ, has non-negative diagonal entries in decreasing order through a post-process of permuting or negating columns of U and/or V; this gives the full SVD of A.[2]

5 Properties of the SVD

One of the first things to see is that if A is SPD (or just semi-definite), the SVD is the same thing as the eigenvalue/eigenvector decomposition

$$A = VDV^T$$

where V is an orthogonal matrix containing the eigenvectors as columns and D is a diagonal matrix with the positive eigenvalues. Obviously, if the eigenpairs are sorted so that the diagonal of D is decreasing, this fits the format of an SVD where $U = V$ and $\Sigma = D$.

Even if A is not symmetric—even if it's rectangular—we can also take a look at the symmetric matrices A^TA and AA^T, based on the SVD of A. For example,

$$\begin{aligned} A^TA &= (U\Sigma V^T)^T(U\Sigma V^T) \\ &= V\Sigma^TU^TU\Sigma V^T \\ &= V\Sigma^T\Sigma V^T \end{aligned}$$

Here we used the fact that U is orthogonal, so $U^TU = I$. This spells out that the singular vectors in V are eigenvectors of A^TA (be careful though: it doesn't necessarily go the other way; an arbitrary eigenvector of A^TA might not be a singular vector of A). Furthermore, the squares of the singular values (from the diagonal square matrix $\Sigma^T\Sigma$) are eigenvalues of A^TA; if A and hence Σ is wider than it is tall then A^TA might have a few additional zero eigenvalues as well. Similar arguments work out these are the same as the eigenvalues of AA^T, up to extra zeros, and U provides a set of eigenvectors for AA^T.

Another interesting link to eigenvalue problems involves the following symmetric matrix:

$$\begin{pmatrix} 0 & A \\ A^T & 0 \end{pmatrix}$$

[2] In fact, this induction proces does produce a diagonal matrix with non-negative entries in decreasing order, i.e. the form we want, but proving that takes a little more work.

Note that this makes sense even if A is rectangular. The positive eigenvalues of this matrix are the singular values of A; the negative eigenvalues are the negatives of the same singular values. The eigenvectors can be composed from concatenating columns of U and V together, or U and $-V$.

Since U and V are orthogonal, $\|Ax\|_2 = \|U^T AV(V^T x)\|_2$, i.e. $\|Ax\|_2 = \|\Sigma y\|_2$ where $y = V^T x$ has the same 2-norm as x. It's not hard to see then that the matrix 2-norm $\|A\|_2$ is just the first (maximum) singular value σ_1.

The Frobenius norm also has a nice relationship with the SVD:

$$\|A\|_F^2 = \mathrm{tr}(A^T A) = \mathrm{tr}(V\Sigma^T \Sigma V^T) = \mathrm{tr}(\Sigma^T \Sigma)$$

In the last step we used the fact that $\mathrm{tr}(BC) = \mathrm{tr}(CB)$, and that $V^T V = I$. Thus $\|A\|_F = \|\Sigma\|_F$, which is just the 2-norm of the list of singular values:

$$\|A\|_F^2 = \sigma_1^2 + \sigma_2^2 + \ldots$$

If A is nonsingular, then the SVD gives a simple formula for its inverse:

$$A^{-1} = (U\Sigma V^T)^{-1} = V\Sigma^{-1} U^T$$

This is almost the SVD of A^{-1} in fact: U and V are swapped, and the diagonal matrix in the middle is:

$$\Sigma^{-1} = \begin{pmatrix} \frac{1}{\sigma_1} & & \\ & \frac{1}{\sigma_2} & \\ & & \ddots \end{pmatrix}$$

These diagonal entries are in increasing order, but otherwise this is in the form of a proper SVD: the singular values of A^{-1} are the reciprocals of the singular values of A.

Putting some of these facts together, we see right away that the 2-norm condition number of A is $\kappa_2(a) = \|A\|_2 \|A^{-1}\|_2 = \sigma_{max}/\sigma_{min}$, the ratio of the biggest to the smallest singular value.

In the singular case, the SVD gives us a very good way to estimate the rank of a matrix. If the first k singular values are much, much larger than the rest (say, on the order of 10^{15} for double precision arithmetic) then it's probably a good bet that the matrix in fact only has rank k—even if the additional singular values are not quite zero, they're plausibly explained as resulting just from rounding errors perturbing exact zeros. More to the point, blithely treating them as sensible positive values and then inverting the matrix will cause the round-off error to completely dominate (by a factor of 10^{15} or so) what resulted, which will thus be useless.

This leads into the idea that the SVD actually lets us deal robustly with rank-deficient problems. We can define the pseudo-inverse of A, denoted A^+, from the SVD as:

$$A^+ = V\Sigma^+U^T$$

where the pseudo-inverse of the diagonal matrix Σ is also diagonal with entries defined as:

$$\Sigma^+_{ii} = \begin{cases} \frac{1}{\sigma_i} & : \quad \sigma_i > 0 \\ 0 & : \quad \text{otherwise} \end{cases}$$

This agrees with the definition of inverse for the nonzero singular components, but zeroes out the ones that can't be "inverted"—removing them from the problem. If the linear system $Ax = b$ is singular (A^{-1} doesn't exist) but is consistent (there are values of x where $Ax = b$, i.e. b is in the range of A) then A^+b is a solution, and in fact of the infinitely many solutions is the one of smallest 2-norm. By using a small but nonzero tolerance in numerically defining A^+, we can robustly solve this type of problem numerically.

The SVD also comes in handy for solving least squares problems. For a full column rank A the problem $\min \|b - Ax\|_2$ is solved by $x = (A^TA)^{-1}A^Tb$, derived from the normal equations even if this isn't always the best way to compute x. With the SVD of A, this becomes:

$$\begin{aligned} x &= (A^TA)^{-1}A^Tb \\ &= (V\Sigma^T\Sigma V^T)^{-1}(V\Sigma^TU^T)b \\ &= V(\Sigma^T\Sigma)^{-1}V^TV\Sigma^TU^Tb \\ &= V\left[(\Sigma^T\Sigma)^{-1}\Sigma^T\right]U^Tb \end{aligned}$$

The diagonal matrix we have bracketed in the middle is nothing other than Σ^+! Thus $x = A^+b$ is the solution to the least-squares problem. In fact, even if A is not full rank, similar arguments can show A^+b is the minimal norm solution (from the infinitely many) of the least squares problem.

6 Low Rank Approximations

Finally, getting back to PCA, the SVD also provides a solution to the problem of finding an optimal low rank approximation to a matrix: for rank k, simply take the first k columns of U and V along with the leading $k \times k$ submatrix of Σ, or in other words, the first k singular vectors and values:

$$A \approx U_{1\to k}\Sigma_{1\to k}V^T_{1\to k}$$

We can see this connection in several ways. For example, using the fact that U and V are orthogonal (and hence multiplication with them doesn't alter the Frobenius norm of a matrix), write

$$\|A - B\|_F^2 = \|\Sigma - U^T B V\|_F^2$$

Finding a rank k matrix B that approximates A in this sense is equivalent to finding a rank k matrix C (connected to B by $B = UCV^T$) that approximates the diagonal matrix Σ. This is an easier problem to tackle directly. Furthermore, if Σ is rectangular it has some zero rows or columns, which means the optimum low rank approximation will have zeros there as well—so we can focus on just the square case (ignoring the extra zeros if Σ is rectangular).

Let's look at the $k = 1$ special case in detail: any rank 1 matrix can be written as an outer-product xy^T, where x has unit 2-norm. The Frobenius norm of the difference is:

$$\|\Sigma - xy^T\|_F^2 = \sum_{i=1}^{n} \left((\sigma_i - x_i y_i)^2 + \sum_{j \neq i} (-x_i y_j)^2 \right)$$

Differentiating this with respect to y_i and setting it to zero (to find a min) gives

$$2(\sigma_i - x_i y_i)(-x_i) + \sum_{j \neq i} 2(-x_j y_i)(-x_j) = 0$$

This reduces to $\|x\|_2^2 y_i = \sigma_i x_i$, which gives:

$$y = \Sigma x$$

Now looking for the optimal x we have the problem of minizing:

$$\begin{aligned}
\sum_{i=1}^{n} \left((\sigma_i - x_i \sigma_i x_i)^2 + \sum_{j \neq i} (-x_i \sigma_j x_j)^2 \right) &= \sum_{i=1}^{n} \left(\sigma_i^2 (1 - x_i^2)^2 + \sum_{j \neq i} \sigma_j^2 x_i^2 x_j^2 \right) \\
&= \sum_{i=1}^{n} \left(\sigma_i^2 (1 - x_i^2)^2 - \sigma_i^2 x_i^4 + \sum_{j=1}^{n} \sigma_j^2 x_i^2 x_j^2 \right) \\
&= \sum_{i=1}^{n} \sigma_i^2 (1 - 2x_i^2) + \sum_{j=1}^{n} \sigma_j^2 x_j^2 \left(\sum_{i=1}^{n} x_i^2 \right) \\
&= \sum_{i=1}^{n} \sigma_i^2 - \sum_{i=1}^{n} 2\sigma_i^2 x_i^2 + \sum_{j=1}^{n} \sigma_j^2 x_j^2 \\
&= \sum_{i=1}^{n} \sigma_i^2 - \sum_{i=1}^{n} \sigma_i^2 x_i^2 \\
&= \|\Sigma\|_F^2 - \|\Sigma x\|_2^2
\end{aligned}$$

Along the way here we used the condition $\|x\|_2 = 1$. This is minimized by choosing a unit length x which maximizes $\|\Sigma x\|_2$; the obvious answer is $x = (1, 0, \ldots, 0)$ using the fact that the diagonal entries of Σ are in decreasing order. Tracing this back to the original problem gives us the desired result, that the first singular vectors and value give the optimal rank one approximation.

7 LAPACK

Though I didn't explicitly mention it in class, LAPACK provides good routines for computing the SVD, and also friendlier "driver" routines to solve problems such as rank-deficient least squares that under the hood use the SVD.