Eigenvalues and Eigenvectors

John Kormylo

Finding eigenvalues is equivalent to finding the roots of its characteristic polynomial. Since I already have nifty algorithms for both finding roots and generating the characteristic polynomial, this is precisely how I find complex eigenvalues.

Most of the time I am concerned with symmetric matrices and only with the largest eigenvalues and their eigenvectors, in which case I like to use steepest descent. The beauty of this approach is that the line search has a closed form solution.

Largest Eigenvalue

For symmetrical matrix A, and eigenvalue λ and its eigenvector $\mathbf{v}$ satisfy the equation

$$A\mathbf{v} = \lambda\mathbf{v} \tag{1}$$

and therefore

$$\lambda = \frac{\mathbf{v}'A\mathbf{v}}{\mathbf{v}'\mathbf{v}} \tag{2}$$

where $'$ denotes transpose. Finding the largest eigenvalue in A is equivalent to maximizing λ as a function of $\mathbf{v}$.

The gradient is given by

$$\mathbf{g} = \frac{\partial\lambda}{\partial\mathbf{v}} = 2\frac{A\mathbf{v} - \lambda\mathbf{v}}{\mathbf{v}'\mathbf{v}} \tag{3}$$

and has the useful property that $\mathbf{g}'\mathbf{v} = 0$. The steepest descent (or ascent) method uses the update

$$\mathbf{v}(k+1) = \mathbf{v}(k) + s\mathbf{g}(k) \tag{4}$$

for $k = 1, 2, \ldots$ starting from some $\mathbf{v}(0)$. Substituting (4) back into (2) gives us

$$\lambda(k+1) = \frac{\mathbf{v}'(k)A\mathbf{v}(k) + 2s\mathbf{v}'(k)A\mathbf{g}(k) + s^2\mathbf{g}'(k)A\mathbf{g}(k)}{\mathbf{v}'(k)\mathbf{v}(k) + s^2\mathbf{g}'(k)\mathbf{g}(k)} \quad . \tag{5}$$

Setting the partial with respect to s to zero gives us

$$\begin{aligned} 0 &= (s\mathbf{g}'A\mathbf{g} + \mathbf{v}'A\mathbf{g})(s^2\mathbf{g}'\mathbf{g} + \mathbf{v}'\mathbf{v}) - (s^2\mathbf{g}'A\mathbf{g} + 2s\mathbf{v}'A\mathbf{g} + \mathbf{v}'A\mathbf{v})(s\mathbf{g}'\mathbf{g}) \\ &= -s^2(\mathbf{v}'A\mathbf{g})(\mathbf{g}'\mathbf{g}) + s\big((\mathbf{g}'A\mathbf{g})(\mathbf{v}'\mathbf{v}) - (\mathbf{v}'A\mathbf{v})(\mathbf{g}'\mathbf{g})\big) + (\mathbf{v}'A\mathbf{g})(\mathbf{v}'\mathbf{v}) \end{aligned} \tag{6}$$

where the index (k) has been removed for convenience. Obviously s is the solution to a quadratic equation which, in this case, always has two real roots. In case A can have negative eigenvalues, one should plug both solutions back into (5) and use the solution with the largest $|\lambda(k+1)|$.

Because the line search has a closed form solution, we know that $\mathbf{g}(k+1)$ will be perpendicular to $\mathbf{v}(k+1) - \mathbf{v}(k) = s\mathbf{g}(k)$. Consequently, conjugate gradients isn't going to do anything for us as they are already conjugates.

As for initial vector $\mathbf{v}(0)$, almost anything will do. Lately I have been using

$$\mathbf{v}' = \left[\begin{array}{ccccc} 1 & 1/2 & 1/6 & \ldots & 1/n! \end{array}\right]$$

since the eigenvectors I am looking for are decaying exponentials and this $\mathbf{v}(0)$ resembles them while being guaranteed not to equal one.

Remaining Eigenvalues

The idea is to constrain a new eigenvector to be orthogonal to any previously computed eigenvectors for matrix A. However, any transformation on A of of the form TAT' for some matrix T changes the eigenvalue problem to a general eigenvalue problem of the form

$$TAT'\mathbf{v} = \lambda TT'\mathbf{v} \quad . \tag{7}$$

As it turns out, the steepest descent method works just as well for this problem.

If one uses the transformation

$$T = I - \frac{\mathbf{v}\mathbf{d}'}{\mathbf{v}'\mathbf{d}} \tag{8}$$

where $\mathbf{d}$ is a unit vector, one will not only orthogonalize A since $T\mathbf{v} = 0$, one will eliminate a complete row and column from A. Assuming $\mathbf{v}'\mathbf{d} = 1$ for simplicity, one can easily show that

$$TAT' = A - \mathbf{v}\mathbf{r}' - \mathbf{r}\mathbf{v}' + a\mathbf{v}\mathbf{v}' \tag{9}$$

where $\mathbf{r} = A\mathbf{d}$ is the row/column to be eliminated and $a = \mathbf{d}'A\mathbf{d}$ is the diagonal term from that row/column. Similarly, one can show that

$$TT' = TIT' = I - \mathbf{v}\mathbf{d}' - \mathbf{d}\mathbf{v}' + \mathbf{v}\mathbf{v}' \tag{10}$$

which basically consists of $I + \mathbf{v}\mathbf{v}'$ with one row/column removed.

For numerical stability, one should choose $\mathbf{d}$ or sort A and $\mathbf{v}$ so as to maximize $|\mathbf{v}'\mathbf{d}|$.

One can express this as an iterative process of solving

$$A(k)\mathbf{v}(k) = \lambda(k)B(k)\mathbf{v}(k) \tag{11}$$

for $k = 1, 2, 3, \ldots, n$ where

$$A(k+1) = \left[I - \frac{\mathbf{v}(k)\mathbf{d}'(k)}{\mathbf{d}'(k)\mathbf{v}(k)}\right] A(k) \left[I - \frac{\mathbf{d}(k)\mathbf{v}'(k)}{\mathbf{d}'(k)\mathbf{v}(k)}\right] \tag{12}$$

and

$$B(k+1) = \left[I - \frac{\mathbf{v}(k)\mathbf{d}'(k)}{\mathbf{d}'(k)\mathbf{v}(k)}\right] B(k) \left[I - \frac{\mathbf{d}(k)\mathbf{v}'(k)}{\mathbf{d}'(k)\mathbf{v}(k)}\right]' \tag{13}$$

starting from $A(1) = A$ and $B(1) = I$. On the last step we have $A(\mathrm{n})$ and $B(n)$ consisting of scalars and therefore $\lambda(n) = A(n)/B(n)$ and $v(n) = 1$.

It should be noted that for the first $n/2$ iterations it is faster to use

$$B(k) = I + V(k)V'(k) \tag{14}$$

where $V(k)$ is an $(n+1-k) \times (k-1)$ matrix consisting of transformed versions of the previously computed eigenvectors.

To reconstruct the eigenvectors to their full size, one merely solves for the missing terms so as to achieve orthogonality with $\mathbf{v}(k)$ for $k = n-1, n-2, \ldots, 1$. For example, we can write

$$\mathbf{v}'(k) \begin{bmatrix} \alpha_i \\ \mathbf{v_i}(k+1) \end{bmatrix} = 0 \tag{15}$$

(wherever α_i needs to go to replace the missing entry) and solve for α_i. Here $\mathbf{v_i}(k)$ denotes one of these partially reconstructed eigenvectors. These intermediate products are not orthogonal to each other, but rather

$$\mathbf{v_i'}(k) B^{-1}(k) \mathbf{v_j}(k) = 0 \quad \forall i \neq j.$$