Total Least Squares

Leo Dorst

April 23, 2018

Let us talk you through a simple case of total least squares. We want to estimate the best line through a set of points. We assume that the noise in the x and y-coordinates is similar. In that case classical least squares is not applicable (it estimates a line in which the x_i-values are assumed to be exact, and the y-error is minimized).

Let the equation of the line be $\mathbf{x}\cdot\mathbf{n}-\delta=0$, with $\mathbf{n}$ normalized to unity so that δ is interpretable as (minus the oriented) distance of the origin to the line.

The oriented distance of an arbitrary point $\mathbf{p}$ to the line is then

$$\mathbf{p}\cdot\mathbf{n}-\delta.$$

(You can derive that in a drawing: the line has support vector $\mathbf{d}=\delta\mathbf{n}$, and we are interested in the length of the component of $\mathbf{p}-\mathbf{d}$ in the $\mathbf{n}$-direction, which is $(\mathbf{p}-\mathbf{d})\cdot\mathbf{n}=\mathbf{p}\cdot\mathbf{n}-\delta$.)

For total least squares, we try to determine $\mathbf{n}$ and δ to minimize the sum of squared distances to the line:

$$\text{minimize} \quad \sum_{i=1}^{N} (\mathbf{p}_i\cdot\mathbf{n}-\delta)^2 \quad \text{while} \quad \|\mathbf{n}\|=1.$$

We take the partial derivative to δ and set it equal to zero. That gives:

$$0=2\sum_{i=1}^{N} (\mathbf{p}_i\cdot\mathbf{n}-\delta)\,(-1).$$

This can be solved for the optimal δ:

$$\delta_* = \tfrac{1}{N}\sum_{i=1}^{N}\mathbf{p}_i\cdot\mathbf{n} \equiv \bar{\mathbf{p}}\cdot\mathbf{n}.$$

Here we introduced the centroid of the data, the average point $\bar{\mathbf{p}}$. This equation shows that δ_* is precisely such that the best fit line passes through the centroid.

Having established this fact, we can simplify our original puzzle as one involving only the direction of the unit normal vector $\mathbf{n}$:

$$\text{minimize} \quad \sum_{i=1}^{N} \big((\mathbf{p}_i-\bar{\mathbf{p}})\cdot\mathbf{n}\big)^2 \quad \text{while } \|\mathbf{n}\|=1.$$

The expression can be rewritten in terms of a quadratic form:

$$\begin{aligned}
\sum_{i=1}^{N} \big((\mathbf{p}_i-\bar{\mathbf{p}})\cdot\mathbf{n}\big)^2 &= \sum_{i=1}^{N} \big(\mathbf{n}\cdot(\mathbf{p}_i-\bar{\mathbf{p}})\big)\big((\mathbf{p}_i-\bar{\mathbf{p}})\cdot\mathbf{n}\big) \\
&= \mathbf{n}^T\Big(\sum_{i=1}^{N} (\mathbf{p}_i-\bar{\mathbf{p}})\big((\mathbf{p}_i-\bar{\mathbf{p}})^T\Big)\mathbf{n} \\
&= \mathbf{n}^T C\mathbf{n}.
\end{aligned}$$

So, we need to minimize a quadratic expression involving a unit vector $\mathbf{n}$ and a symmetric matrix C (which you may recognize as covariance matrix (times N)). Since the matrix C is symmetric, it can be diagonalized using an orthogonal transformation: *all symmetric matrices are orthogonally diagonalizable* (look it up in your LA course). The diagonalization results in a diagonal matrix Λ with the eigenvalues of C on the diagonal, and the orthogonal matrix Q performing the diagonalization has as columns the eigenvectors of C.

$$C = Q\,\Lambda\,Q^T$$

The detailed construction of C through squares also implies that C is semi-positive-definite: it has no negative eigenvalues (this is harder to show, we give a hint below). So we can order the eigenvalues from big to small, and the smallest may be zero.

Diagonalization transforms the problem to a simpler problem:

$$\begin{aligned}
&\text{minimize}\ \ \mathbf{n}^T C\,\mathbf{n} \quad \text{while} \quad \|\mathbf{n}\| = 1 \\
&\qquad\qquad \Leftrightarrow \\
&\text{minimize}\ \ \mathbf{n}^T Q\,\Lambda\,Q^T\mathbf{n} \quad \text{while} \quad \|\mathbf{n}\| = 1 \\
&\qquad\qquad \Leftrightarrow \\
&\text{minimize}\ \ \mathbf{m}^T \Lambda\,\mathbf{m} \quad \text{while} \quad \|\mathbf{m}\| = \|Q^T\mathbf{n}\| = \|\mathbf{n}\| = 1
\end{aligned}$$

You can write out the diagonalized expression as (we give it for N-D, although for us working on lines in the plane, N is obviously equal to 2):

$$\text{minimize}\ \ \lambda_1 m_1^2 + \cdots \lambda_N m_N^2 \ \ \text{while}\ \ m_1^2 + m_2^2 + \cdots + m_M^2 = 1.$$

The diagonalized expression has the same values as the original expression - it is merely a rewriting. But constrained minimization of the diagonalized expression is much easier: if you have to choose the coefficients m_i such that the sum of the squares equals 1, and the expression is as small as possible, the best things to do is make the $m_N = 1$ for the smallest eigenvalue λ_N and $m_i = 0$ for the rest; anything else would increase the sum. (Later on you may encounter the method of *Lagrange multipliers* to solve a problem like this without such a reasoning step.)

So the solution is $\mathbf{m}_* = [0\ 0\ \cdots\ 0\ 1]^T$ (in our 2-D case, $\mathbf{m}_* = [0\ 1]^T$,) and that gives for the unit normal vector $\mathbf{n}_*$ of the optimal hyperplane:

$$\mathbf{n}_* = Q\,\mathbf{m}_* = \text{normalized eigenvector } \mathbf{v}_N \text{ of } C \text{ for smallest eigenvalue } \lambda_N$$

By the way, the corresponding λ_N is the 'error' of the best fit, the sum of the squares of the perpendicular distances to the fitted hyperplane (or line in our 2D case). This can be zero: that would be the exact fit. (This is a clue towards showing that the matrix C is semi-definite-positive: teach of the λ_i can be seen as the sum of squared distances to the plane with as normal the corresponding eigenvector $\mathbf{v}_i$, passing through $\bar{\mathbf{p}}$, and thus all eigenvalues are non-negative.)

So with δ_* and $\mathbf{n}_*$ known, we have determined the optimal hyperplane, passing as well as possible 'through' the data in the TLS sense.

In Matlab, it is handy to make a matrix of which the i-th column is the data point $\mathbf{p}_i$; then average all the columns to obtain $\bar{p}$; subtract that off each column to make a 'relative data point matrix' P. Then $C = PP^T$ (verify that!), and you need to use `eig()` (or `svd()`, though that is overkill) to get its eigenstructure.