ELIMINATION METHODS FOR LEAST SQUARES

David F. Gleich
August 21, 2023

1 ELIMINATION METHODS FOR LEAST SQUARES

1.1 THE SIMPLE ELIMINATION SOLVE

We can also use variable elimination for least squares problems. Consider

$$\text{minimize} \quad \|\boldsymbol{A}\mathbf{x} - \mathbf{b}\|.$$

Partition $\boldsymbol{A} = \begin{bmatrix} \mathbf{a} & \boldsymbol{C} \end{bmatrix}$ and $\mathbf{x} = \begin{bmatrix} \gamma \\ \mathbf{y} \end{bmatrix}$. Then

$$\text{minimize} \quad \|\gamma \mathbf{a} + \boldsymbol{C}\mathbf{y} - \mathbf{b}\|.$$

We proceed as follows, suppose we know $\mathbf{y}$. Let $\mathbf{d} = \boldsymbol{C}\mathbf{y} - \mathbf{b}$. Then this is just the one varible least squares problem

$$\text{minimize} \quad \|\gamma \mathbf{a} - \mathbf{d}\|.$$

If we explain what this is, then we are looking for the best *scaling* of the vector $\mathbf{a}$ to get us as close to possible to $\mathbf{d}$.

TODO Add figure that explains this

A little bit of thinking yields the following insight: the scaling of $\mathbf{a}$ is *closest* to $\mathbf{d}$ when the difference $\gamma\mathbf{a} - \mathbf{d}$ is *orthogonal* to $\mathbf{a}$. If this weren't the case, then we could decrease the distance by moving a little bit in any direction. Hence, the solution γ must satisfy the relationship:

$$\mathbf{a}^T(\gamma \mathbf{a} - \mathbf{d}) = 0 \quad \text{or} \quad \gamma = \frac{1}{\mathbf{a}^T\mathbf{a}}\mathbf{a}^T\mathbf{d}.$$

Now, we proceed as follows and substitute $\gamma(\mathbf{y})$ into our original least squares problem

$$\text{minimize} \quad \|\gamma(\mathbf{y})\mathbf{a} + \boldsymbol{C}\mathbf{y} - \mathbf{b}\| \quad \to \quad \text{minimize} \quad \|\tfrac{1}{\mathbf{a}^T\mathbf{a}}\mathbf{a}^T(\boldsymbol{C}\mathbf{y} - \mathbf{b})\mathbf{a} + \boldsymbol{C}\mathbf{y} - \mathbf{b}\|.$$

We can simplify this expression to

$$\text{minimize} \quad \|(\boldsymbol{I} - \tfrac{1}{\mathbf{a}^T\mathbf{a}}\mathbf{a}^T\mathbf{a}^T)(\boldsymbol{C}\mathbf{y} - \mathbf{b})\|.$$

This new problem has one fewer variable. If we recurse on this idea, we have the following algorithm.

```
function least_squares_eliminate(A,b)
  a = A[:,1]
  na = norm(a)
  q = a/na
  if size(A,2) == 1
    return [a'*b]/na
  end
  y = least_squares_eliminate(A[:,2:end]-q*q'*A[:,2:end], b - q*q'*b)
  γ = q'*(b - A[:,2:end]*y)/na
  return pushfirst!(y,γ)
end
```

1.2 A MATRIX VERSION

TODO See if we can get something better here…

The matrix structure in this problem is already slightly apparent. Let $\boldsymbol{T} = (\boldsymbol{I} - \frac{1}{\mathbf{a}^T\mathbf{a}}\mathbf{a}^T\mathbf{a}^T)$. Then we have

$$\text{least-squares}(\boldsymbol{A}, \mathbf{b}) \to \text{least-squares}(\boldsymbol{TAS}, \boldsymbol{T}\mathbf{b}).$$

Here $\boldsymbol{S}$ is a matrix that selects the last $n-1$ columns of a matrix.

Now, it turns out there is an issue here. The matrix $\boldsymbol{T}$ is a special type of matrix called a *projection*. A projection matrix is any matrix where $\boldsymbol{T}^2 = \boldsymbol{T}$. It represents a projection onto a subspace, so $\boldsymbol{T}^2 = \boldsymbol{T}$ because the projection of a projection is the same projection. For this matrix $\boldsymbol{T}$ it's just a few lines of algebra to verify that $\boldsymbol{T}^2 = \boldsymbol{T}$.

TODO Add these lines

This is a small issue, though, because $\boldsymbol{T}\mathbf{a} = 0$ and so $\boldsymbol{TA} = \begin{bmatrix} 0 & \boldsymbol{TC} \end{bmatrix}$. Thus, we lose all the information associated with $\mathbf{a}$ after the first transformation. However, suppose we just *memorize this* and store $\mathbf{a}$ – after normalization – at each iteration into a matrix $\boldsymbol{Q}$.

To be entirely precise, let $\boldsymbol{A} = \begin{bmatrix} \mathbf{a}_1 & \cdots & \mathbf{a}_n \end{bmatrix}$. Let $\boldsymbol{T}_1, \ldots \boldsymbol{T}_n$ be the matrix $\boldsymbol{I} - \mathbf{q}_i\mathbf{q}_i^T$ formed in the least squares elimination algorithm at the ith call. Then we have:

$$\mathbf{q}_i = \frac{1}{\|\boldsymbol{T}_{i-1} \cdots \boldsymbol{T}_1 \mathbf{a}_i\|} \boldsymbol{T}_{i-1} \cdots \boldsymbol{T}_1 \mathbf{a}_i.$$

In a bit of remarkable luck, the matrix $\boldsymbol{Q}$ turns out to be orthogonal. In fact, it's the result of the Gram-Schmidt process.

1.3 THE GRAM-SCHMIDT PROCESS

TODO

1.4 THE QR FACTORIZATION

All of these ideas can be generalized. The idea is that we transform $\boldsymbol{A}$