Least Squares Problems
LSQ + Householder Triangularization
Conditioning
Numerical Matrix Analysis
Lecture Notes #15 — Conditioning and Stability

Peter Blomgren,
⟨blomgren.peter@gmail.com⟩
Department of Mathematics and Statistics
Dynamical Systems Group
Computational Sciences Research Center
San Diego State University
San Diego, CA 92182-7720
http://terminus.sdsu.edu/

Spring 2016

Outline

1 Least Squares Problems
   - Recap: Conditioning
   - Recap: Solution Strategies
   - Experiment: Test Problem

2 LSQ + Householder Triangularization
   - Theorem
   - Relative Error
   - Comparison with Gram-Schmidt

3 Conditioning
   - Normal Equations vs. Householder QR?!?
   - The SVD
   - Comments & Rank-Deficient Problems

---

Last Time

Theorem (Conditioning of Linear Least Squares Problems)

Let \( \bar{b} \in \mathbb{C}^m \) and \( A \in \mathbb{C}^{m \times n} \) of full rank be given. The least squares problem, \( \min_{\bar{x} \in \mathbb{C}^n} \| \bar{b} - A\bar{x} \| \) has the following 2-norm relative condition numbers describing the sensitivities of \( \bar{y} = PB \in \text{range}(A) \) and \( \bar{x} \) to perturbations in \( \bar{b} \) and \( A \):

<table>
<thead>
<tr>
<th>Input</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \bar{b} )</td>
<td>( \frac{1}{\cos \theta} )</td>
</tr>
<tr>
<td>( A )</td>
<td>( \frac{\kappa(A)}{\cos \theta} )</td>
</tr>
</tbody>
</table>

\( \kappa(A) = \frac{\sigma_1}{\sigma_n} \in [1, \infty) \),
\( \cos \theta = \frac{\| \bar{y} \|}{\| \bar{b} \|} \in \left[ 0, \frac{\pi}{2} \right] \),
\( \eta = \frac{\| A \| \| \bar{x} \|}{\| A\bar{x} \|} \in [1, \kappa(A)] \)

---

Deconstructing \( \eta \)... 

\[ \eta = \frac{\| A \| \| \bar{x} \|}{\| A\bar{x} \|} \in [1, \kappa(A)] \]

Without loss of generality, rescale \( \bar{x} \) so that \( \| \bar{x} \| = 1 \).

Now with \( A = U\Sigma V^* \), the extreme cases correspond to

\[ \bar{x} = \bar{v}_1 \sim \eta = \frac{\| A \|}{\| A\bar{v}_1 \|} = \frac{\sigma_1}{\sigma_1} = 1, \]

\[ \bar{x} = \bar{v}_n \sim \eta = \frac{\| A \|}{\| A\bar{v}_n \|} = \frac{\sigma_1}{\sigma_n} = \kappa(A). \]

So, we get the best conditioning of the Least Squares Problem when the formulation and model conspires such that the final solution aligns with the minor semi-axis of the ellipsoid \( AS^{n-1} \).
Solving Least Squares Problems — 4 Approaches

Currently, we have four candidate methods for solving least squares problems:

- **The Normal Equations**
  \[ \bar{x} = (A^*A)^{-1}A^*\bar{b} \]

- **Gram-Schmidt Orthogonalization** (QR-factorization)
  \[ \bar{x} = R^{-1}(Q^*\bar{b}) \]

- **Householder Triangularization** (QR-factorization)
  \[ \bar{x} = R^{-1}(Q^*\bar{b}) \]

- **The Singular Value Decomposition**
  \[ \bar{x} = V(\Sigma^{-1}(U^*\bar{b})) \]

Our Test Problem

```matlab
% The Dimensions of the problem
m = 100;
n = 15;

% The time-vector --- samples in [0,1]
t = (0:(m-1))’ / (m-1);

% Build the matrix A
A = [];
for p = 0:(n-1)
    A = [ A t.^p ];
end

% Build the right-hand side
b = exp(sin(4*t)) / 2006.787453080206;
```

Our Test Problem: Visualized

![Graph showing the rows of the matrix A, the columns of the matrix A, and the vector b.](image)

2006.787453080206 ???

The normalization

```matlab
% Build the right-hand side
b = exp(sin(4*t)) / 2006.787453080206;
```

Is chosen so that the correct (exact) value of the last component is \( x_{15} = 1 \).

We are trying to compute the 14th degree polynomial \( p_{14}(t) \) which fits \( \exp(\sin(4t)) \) on the interval \([0,1]\).
Finding 2006.787453080206 — Using Maple

Some Maple Action...

```
with(linalg);
Digits := 512;
m := 100;
n := 15;
f := (i,j) -> ((i-1)/(m-1));
g := (i) -> exp(sin(4*(i-1)/(m-1)));
b := Vector(100,g);
x := leastsqr(A,b);
evalf(x[15]);
```

Gives

\[ x_{15} = 2006.7874531048518338 \ldots \]

Curious... However, using this value instead didn’t change anything significantly in the following slides...

Associated Condition Numbers

We use the best available solution \( x = A\backslash b; \ y = A\ast x; \) to estimate the dimensionless parameters, and condition numbers

<table>
<thead>
<tr>
<th>( \kappa(A) )</th>
<th>( \cos \theta )</th>
<th>( \eta )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \text{cond}(A) )</td>
<td>( \text{norm}(y)/\text{norm}(b) )</td>
<td>( \text{norm}(A) \ast \text{norm}(x)/\text{norm}(y) )</td>
</tr>
<tr>
<td>( 2.27 \times 10^{10} )</td>
<td>( 0.9999999999426 )</td>
<td>( 2.10 \times 10^{5} )</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>( \downarrow \text{Input, Output} \rightarrow )</th>
<th>( \bar{y} )</th>
<th>( \bar{x} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \bar{b} )</td>
<td>1.00</td>
<td>1.08 \times 10^{5}</td>
</tr>
<tr>
<td>( A )</td>
<td>( 2.27 \times 10^{10} )</td>
<td>( 3.10 \times 10^{10} )</td>
</tr>
</tbody>
</table>

**Bottom Line:** If we get 6 correct digits (error \( \sim 10^{-6} \)) in matlab (\( \epsilon_{\text{mach}} \sim 10^{-10} \)) then we are doing as well as we can.

Householder Triangularization

We have three ways of solving the least squares problem using Householder Triangularization

- In the first approach, we explicitly form and use the matrix \( Q \).
- In the second approach, we extract the “action” \( Q^\ast \bar{b} \), by adding \( \bar{b} \) as an additional column in \( A \), and then identifying the appropriate components of the computed \( R \) as \( R \) and \( Q^\ast \bar{b} \).
- In the third approach, we rely on matlab’s implementation... It uses Householder triangularization with column pivoting, for maximal accuracy.

The approaches described above gives us the following errors

\[ e_1 = 3.16387 \times 10^{-7}, \ e_2 = 3.16371 \times 10^{-7}, \ e_3 = 2.18674 \times 10^{-7} \]

Implicitly forming \( Q^\ast \bar{b} \) improves the result marginally, which means that the errors introduced in the explicit formation of \( Q^\ast \bar{b} \) are small compared to the errors introduced by the QR-factorization itself.

The Matlab solver, which includes all the bells and whistles, improves the result a little more;

All three variants are backward stable.
Householder Triangularization: Theorem

Let the full-rank least square problem be solved by Householder triangularization in a floating-point environment satisfying the floating point axioms. This algorithm is backward stable in the sense that the computed solution $\tilde{x}$ has the property

$$
\| (A + \delta A)\tilde{x} - \tilde{b} \| = \min_{\| \delta A \| = O(\epsilon_{\text{mach}})} \| \delta A \| = O(\epsilon_{\text{mach}})
$$

for some $\delta A \in \mathbb{C}^{m \times n}$. This is true whether $\tilde{Q}^*\tilde{b}$ is formed explicitly or implicitly. Further, the theorem is true for Householder triangularization with arbitrary column pivoting.

Modified Gram-Schmidt Orthogonalization

We have two ways of solving the least squares problem using modified Gram-Schmidt orthogonalization

\begin{verbatim}
[Q,R] = qr_mgs(A);
x = R(1:n,n+1);
e4 = abs(x(15)-1);
\end{verbatim}

\begin{verbatim}
[Q,R] = qr_mgs([A b]);
Qb = R(1:m,n+1);
R = R(1:n,1:n);
x = R \backslash Qb;
e5 = abs(x(15)-1);
\end{verbatim}

- The explicit formation of $Q$ in the first approach suffers from forward errors, and the result is quite disastrous

  $$
e_4 = 0.03024
$$

- If instead we form $Q^*\tilde{b}$ implicitly (the second approach), the result is much better

  $$
e_5 = 2.4854 \times 10^{-8}
$$

Modified Gram-Schmidt Orthogonalization: Comments and Theorem

The fact that $e_5 < e_{1,2,3}$ in this example is not an indication of anything in particular — it is just luck.

The following is a provable result:

Theorem

The solution of the full-rank least squares problem by modified Gram-Schmidt orthogonalization is also backward stable, provided that $Q^*b$ is formed implicitly, as indicated on the previous slide.
Normal Equations

Even though the condition number for the least squares problem is

$$\kappa(A) + \frac{\kappa(A)^2 \tan \theta}{\eta}$$

we have successfully found the solution with ~ 6 correct digits.

It may seem like it would be OK to find the solution via the **normal equations**

$$\tilde{x} = (A^*A)^{-1}(A^*b)$$

since

$$\kappa(A^*A) \sim \kappa(A)\kappa(A^*) \sim \kappa(A)^2$$

However, if we try this ($x = (A'*A)\backslash(A'*b)$) in matlab, we get

Warning: Matrix is close to singular or badly scaled. Results may be inaccurate. RCOND = 1.512821e-19.

and $$|\tilde{x}_{15} - x_{15}| = 1.678.$$ 

Normal Equations: Theorem

**Theorem**

The solution of the full-rank least squares problem via the normal equations is **unstable**. Stability can be achieved, however, by restriction to a class of problems in which \( \kappa(A) \) is uniformly bounded above or \( \frac{\tan \theta}{\eta} \) is uniformly bounded below.

**Bottom Line:** The normal equations only work for “easy” least squares problems.

Normal Equations: What Happened?!?

Even though the worst-case conditioning for the least squares problem is \( \kappa(A)^2 \), that is almost never realized.

In our test problem

$$\tan \theta \sim 3 \times 10^{-6}, \quad \eta \sim 2 \times 10^5$$

so, whereas

$$\kappa(A)^2 = 5.16 \times 10^{20}, \quad \frac{\kappa(A)^2 \tan \theta}{\eta} = 3.10 \times 10^{10}$$

But for \( A^*A \) there are no mitigating factors, and

$$\kappa(A^*A) = 2.0 \times 10^{18}$$

underestimate?

so

$$\kappa(A^*A) \cdot \epsilon_{\text{mach}} = 4.4 \times 10^2$$

The Singular Value Decomposition

Solving the least squares problem using the SVD is the most expensive, but also the most stable method; here we get our error to be of the same order of magnitude as the other stable methods

$$e_6 = 3.16383 \times 10^{-7}$$

**Theorem**

The solution of the full-rank least squares problem by the SVD is **backward stable**.
At this point we have four working backward stable approaches to solving the full rank least squares problem:

- Householder triangularization
- Householder triangularization with column pivoting
- Modified Gram-Schmidt with implicit $Q^*\bar{b}$ calculation
- The SVD

The differences, in terms of classical norm-wise stability, among these algorithms are minor.

For everyday use, select the simplest one — Householder triangularization — as your default algorithm. If you are working in matlab use $A\backslash\bar{b}$ — Householder triangularization with column pivoting.

When $\text{rank}(A) < n$, quite possibly with $m < n$, the least squares problem is **under-determined**.

No unique solution exists, unless we add additional constraints. Usually, we look for the **minimum norm** solution $\bar{x}$; i.e. among the solutions we select the one with smallest norm.

The solution depends (strongly) on $\text{rank}(A)$, and determining numerical rank is non-trivial. Is $10^{-14} = 0$???

For this class of problems, the only fully stable algorithms are based on the SVD.

Householder triangularization with column pivoting is stable for “almost all” such problems.

Rank-deficient least squares problems are a completely different class of problems, and we sweep all the details under the rug...