Stability of matrix products

Analogously to what we did for the scalar product, one can bound the error in the product of two matrices $C = AB$, with $|\widetilde{C} - C| \leq n|A||B|\mathrm{u} + O(\mathrm{u}^2)$.

Passing to norms,

$$\|\widetilde{C} - C\| \leq \mathcal{O}(\mathrm{u})\|A\|\|B\|$$

Here, $\mathcal{O}(\mathrm{u})$ contains polynomial factors $\mathcal{O}(n)$, $\mathcal{O}(n^2)$.... The exact degree depends on the choice of the norm.

This is not a small relative error: $\|A\|\|B\|$ can be much larger than $\|C\|$.

The computed result is not backward stable: indeed,

$$\widetilde{C} = AB + F, \quad \|F\| \leq \mathcal{O}(\mathrm{u})\|A\|\|B\|$$

becomes (if A is square and invertible)

$$\widetilde{C} = A(\underbrace{B + A^{-1}F}_{:=\widehat{B}}), \quad \|\widehat{B} - B\| = \|A^{-1}F\| \leq \mathcal{O}(\mathrm{u})\|A^{-1}\|\|A\|\|B\|.$$

Backward stability and orthogonal transformations

However, the result is stable in an important case: if $A = Q$ is orthogonal; indeed in that case $\|A\|_2 = \|A^{-1}\|_2 = 1$.

Backward stability of orthogonal transformations

If $Q \in \mathbb{R}^{m \times m}$ is orthogonal and $B \in \mathbb{R}^{m \times n}$, then the computed version $\widetilde{C}$ of $C = QB$ is backward stable:

$$\widetilde{C} = Q\widehat{B}, \quad \|\widehat{B} - B\| \leq \mathcal{O}(\mathrm{u})\|B\|.$$

The same result holds if QB is computed using Householder transformations: a bound for the forward error F can be obtained via stability analysis of computing $\mathbf{u}$ and $HB = B - 2\mathbf{u}\mathbf{u}^T B$ (boring algebra), but then the rest is the same.

Stability of QR factorization

Now we wish to prove a backward stability result for the QR factorization, $\tilde{Q}\tilde{R} = \mathtt{qr}(A + \Delta_A)$.

Recall: in QR factorization, an upper triangular $R = R_n$ is obtained after n steps of the form

$$R_0 = A, \quad \underbrace{\begin{bmatrix} I & \\ & H(\mathbf{u}_k) \end{bmatrix}}_{Q(\mathbf{u}_k)} \underbrace{\begin{bmatrix} * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & * & * & * \end{bmatrix}}_{R_{k-1}} = \underbrace{\begin{bmatrix} * & * & * & * & * \\ 0 & * & * & * & * \\ 0 & 0 & * & * & * \\ 0 & 0 & 0 & * & * \\ 0 & 0 & 0 & * & * \end{bmatrix}}_{R_k}, \quad k = 1, 2, \ldots, n.$$

Backward stability of QR

In machine arithmetic, the computed $\tilde{R}$ and the (exactly orthogonal) $\tilde{Q} = \tilde{Q}_1 \cdots \tilde{Q}_n = Q(\tilde{\mathbf{u}}_1) \cdots Q(\tilde{\mathbf{u}}_n)$ satisfy

$$\tilde{Q}\tilde{R} = \hat{A} = A + \Delta, \quad \|\Delta\| = \mathcal{O}(\mathrm{u})\|A\|.$$

Note: $\tilde{Q}$ is defined implicitly through the $\mathbf{u}_k$.

Proof

We replay each step and turn the errors into perturbations of A:

$$\tilde{R}_0 = A$$

$$\tilde{R}_1 = \tilde{Q}_1 \odot \tilde{R}_0 = \tilde{Q}_1(A + \Delta_1), \quad \|\Delta_1\| = \mathcal{O}(\mathrm{u})\|\tilde{R}_0\|.$$

$$\begin{aligned} \tilde{R}_2 = \tilde{Q}_2 \odot \tilde{R}_1 &= \tilde{Q}_2(\tilde{R}_1 + \Delta_2) = \tilde{Q}_2(\tilde{Q}_1(A + \Delta_1) + \Delta_2) \\ &= \tilde{Q}_2\tilde{Q}_1(A + \Delta_1 + \tilde{Q}_1^T\Delta_2), \quad \|\Delta_2\| = \mathcal{O}(\mathrm{u})\|\tilde{R}_1\|. \\ &\vdots \end{aligned}$$

At each step we have a new perturbation with norm

$$\|Q_{k-1}^T \cdots Q_1^T \Delta_k\| = \|\Delta_k\| = \mathcal{O}(\mathrm{u})\|\tilde{R}_{k-1}\| = \mathcal{O}(\mathrm{u})(\|A\| + \mathcal{O}(\mathrm{u})).$$

Since all transformations are orthogonal, all n error terms are bounded by $\mathcal{O}(\mathrm{u})\|A\|$.

Backward stability of least squares algorithms

With more work, we can extend the reasoning to cover the whole LS solution:

```
function x = solve_ls_QR(A, y)
[Q1, R1] = qr(A, 0); %backward stable
c = Q1'*y; %backward stable
x = R1 \ c; %backward stable
```

Backward stable + orthogonal transformations: all errors $O(\mathrm{u})\|A\|$.

Similarly,

```
function x = solve_ls_SVD(A, y)
[U, S, V] = svd(A, 0); %backward stable
c = U'*y; %backward stable
d = c ./ diag(S); %backward stable
x = V*d; %backward stable
```

Backward stable + orthogonal transformations: all errors $O(\mathrm{u})\|A\|$.

The problem with normal equations

```
function x = solve_ls_NE(A, y)
C = A' * A;
d = A' * y;
x = C \ d;
```

Not backward stable: the transformation $A\mathbf{x} - \mathbf{y} \to A^T(A\mathbf{x} - \mathbf{y})$ is not orthogonal, and may convert our problem into a more ill-conditioned one!

Indeed, even if we just consider the machine arithmetic error in storing $\tilde{\mathbf{d}}$, solving the system may amplify it by a factor $\kappa(A^TA) = \kappa(A)^2$.
TL;DR (we did not give full proofs)

The QR and SVD methods are backward stable, but normal equations produce errors of size $\kappa(A)^2$ even when the conditioning of the LS problem is smaller.

Comparison of least squares algorithms

	Normal eqns	QR	SVD
$m \approx n$	$\frac{4}{3}n^3$	$\frac{4}{3}n^3$	$\approx 13n^3$
$m \gg n$	mn^2	$2mn^2$	$2mn^2$
	Unstable for small θ	Backward stable	Backward stable; reveals distance from singularity, allows regularization

Know when to use each one. QR is a good 'generic' choice.

A priori error bound on x

QR and SVD (but not NE!) always deliver a solution $\tilde{\mathbf{x}} = \mathbf{x} + \mathbf{e}$ with

$$\|\mathbf{e}\| \leq O(\mathrm{u})(\kappa_{rel,A\to\mathbf{x}} + \kappa_{rel,\mathbf{y}\to\mathbf{x}})\|\mathbf{x}\|.$$

Exercises

1. Can you explain our earlier example in which normal equations delivered a wrong result, in view of this theory? Are those errors what you would expect in theory? Is that example in the case $\theta \approx 0$, $\theta \approx 90^\circ$, or in the general case?
2. Show that $\|A^T A\| = \|A\|^2$. Hint: recall how $\|A\| = \|U\Sigma V^T\| = \|\Sigma\| = \sigma_1$: can we do something similar for $A^T A$?
3. Show that $\kappa(A^T A) = \kappa(A)^2$.

Book references Trefethen-Bau, Lectures 16, 19. Higham, *Accuracy and Stability of Numerical Algorithms*, Chapters 19 and 20, for exact bounds with all the coefficients worked out instead of big-Os.