Drake Brown
Date Apr 14, 2025

Numerical Methods for Linear Algebra 1234

1 Numerical Stability

Historically, this has been a very difficult topic for me to understand. Here is how I will put it.

In floating-point arithmetic, operations such as addition, multiplication, rounding, and exponentiation are not exact, and are often modeled as

$$\text{fl}(x) = x(1+\epsilon), \text{ rounding error}$$

where ϵ is a small error term (typically on the order of machine epsilon). For example, the floating-point evaluation of $x+1$ might be modeled as

$$x \oplus 1 = (x+1)(1+\epsilon)$$
$$\text{fl}(x) \otimes 1 = (x(1+\epsilon_1)+1)(1+\epsilon_2)$$

When analyzing floating-point computations, we often use $\tilde{f}(x)$ to denote the computed version of a function $f(x)$. However, this can mean one of two things, depending on the context:

1. **Full error tracking**: We include rounding errors from both the inputs and each intermediate operation. For instance, for $f(x) = x^2$, we might write

$$\tilde{f}(x) = (x(1+\epsilon_1))^2 (1+\epsilon_2).$$

2. **Operation-only error tracking**: We assume the input is exact and only account for errors from operations. In this case, the same function might be modeled as

$$\tilde{f}(x) = (x^2)(1+\epsilon).$$

For simple functions, it's often feasible to track all rounding errors explicitly. But in more complex expressions or algorithms, we typically focus on operation-induced errors and omit the initial input error for simplicity.

We also often denote $\tilde{x} = x + \delta x$

There are two things that we are concerned about with this topic: stability and backwards stability.

A function f is backwards stable if:

$$\tilde{f}(x) = f(\tilde{x}) = f(x + \delta x) \tag{1}$$

$$\text{for some } \delta x \text{ with :} \tag{2}$$

$$\frac{\|\tilde{x} - x\|}{\|x\|} = \frac{\|\delta x\|}{\|x\|} = O(\epsilon) \tag{3}$$

Ofsten the steps to proving this was to take $\tilde{f}(x)$ and try to find some δx such that $\tilde{f}(x) = f(x+\delta x)$. This is done by expanding all of the operations using operation-only error tracking (So it should be in terms of ϵ_ξ). One then takes that computed δx and plugs it into the second condition to show it is $O(\epsilon)$.

A function f is stable if we relax the first condition:

$$\frac{\left\|\tilde{f}(x) - f(\tilde{x})\right\|}{\|f(\tilde{x})\|} = O(\epsilon) \tag{4}$$

$$\text{for some } \delta x \text{ with :} \tag{5}$$

$$\frac{\|\tilde{x} - x\|}{\|x\|} = \frac{\|\delta x\|}{\|x\|} = O(\epsilon) \tag{6}$$

1.1 Stability of unitary matrix multiplication

A non standard way to prove backwards stability in this case is to notice that if an algorithm is backwards stable:

$$\tilde{QA} = QA + Q\Delta A \tag{7}$$

$$\|\delta A\| = \|Q\Delta A\| = \left\|\tilde{QA} - QA\right\| \tag{8}$$

Where we used both linearity and unitarity. Typically when we are given a matrix, we show component wise backwards stability. This is much stronger than norm stability because the sum of all of the entries in the matrix is certainly greater than the infinity or one norm, the statement then follows from norm equivalence.

From here I will note without proof that for $f(x) = q^*x$ we have $\tilde{f}(x) = q^*x + mq^*xO(\epsilon)$. We then note that the i, jth entry of $\tilde{QA} - QA$ is then:

$$mQ_i^* A_{:,j} O(\epsilon) \tag{9}$$

since Q is unitary we know its entries are all less than 1 so:

$$\leq mO(\epsilon) \sum_{k=0}^{m} A_{k,j} \tag{10}$$

$$\leq m \|A\| O(\epsilon) \tag{11}$$

We can bound the norm of the whole matrix by adding together all of the entries (corresponding to multiplying the above by m^2). Thus we get backwards stability:

$$\frac{\|\delta A\|}{\|A\|} \leq \frac{m^3 \|A\| O(\epsilon)}{\|A\|} = m^3 O(\epsilon) = O(\epsilon) \tag{12}$$

This tells us that computing the householder QR decomposition is numerically stable.

1.2 Stability of QR solver

Consider the problem $Ax = b$ and we want to solve for x. To do this we can apply the QR decomposition $QRx = b$ apply Q^* on both sides $Rx = Q^*b$ and then use backsubstitution to solve for things. Note that if we compute the QR decomposition using housholder transformations, that amounts to multiplication by a unitary matrix at each step. Thus obtaining the QR decomposition and solving for Q^*b is actually backwards stable by what we just proved. The book also proves that backsubstitution is stable (I will not go into details)

To prove that the algorithm above is backwards stable consider:

$$\tilde{f}(A) = f(\tilde{A}) \tag{13}$$

$$(\tilde{Q} + \delta Q)(\tilde{R} + \delta R)x = (A + \Delta A)\tilde{x} \tag{14}$$

$$(\tilde{Q}\tilde{R} + (\delta Q)\tilde{R} + \tilde{Q}(\delta R) + (\delta Q)(\delta R))x \tag{15}$$

$$= (A + \delta A + (\delta Q)\tilde{R} + \tilde{Q}(\delta R) + (\delta Q)(\delta R))x \tag{16}$$

So those four latter terms are ΔA and we need to show that $\frac{\|\Delta A\|}{\|A\|} = O(\epsilon)$