Methods for general matrix functions

We now explore methods for matrix functions in general (not restricting to specific choices of f). [Higham book, Ch. 4]

Simplest strategy (if A diagonalizable): $A = V\Lambda V^{-1}$, then

$$f(A) = Vf(\Lambda)V^{-1} = V \begin{bmatrix} f(\lambda_1) & & \\ & \ddots & \\ & & f(\lambda_m) \end{bmatrix} V^{-1}.$$

Works fine if A is symmetric/Hermitian/normal (and Q orthogonal). Otherwise, errors on $f(\lambda_i)$ (or in the diagonalization itself) are amplified by a factor $\kappa(V)$ — possibly much higher than the conditioning of the problem.

Example: sqrt of $\left[\begin{smallmatrix} 3 & -1 \\ 1 & 1 \end{smallmatrix}\right]$: Matlab computes an eigenvector matrix V with $\kappa(V) \approx 10^7$, and computing $f(A)$ via diagonalization 'loses' 7 significant digits with respect to the exact result (which you can compute with the interpolating polynomial).

Polynomial evaluation

How to evaluate polynomials in a matrix argument?

- Direct evaluation: compute powers of X by successive products, take a linear combination of them).
- Horner method: $(\dots(((a_d X + a_{d-1})X + a_{d-2})X + \dots)X + a_0 I$

Bulk of the cost: $d-1$ matrix products, in both cases. Unlike the scalar case, the two methods are essentially equivalent in terms of cost.

Cheaper: divide the terms into 'chunks' of size approx. $\sqrt{d}$, e.g.,

$$(p_8 A^2 + p_7 A + p_6)(A^3)^2 + (p_5 A^2 + p_4 A + p_3)A^3 + (p_2 A^2 + p_1 A_1 + p_0).$$

This is known as Paterson-Stockmayer method. Fewer multiplications, but requires more storage.

Stability of polynomial evaluation methods

All these polynomial evaluation methods are stable only with respect to the 'absolute value' polynomial.

Theorem

The value $\tilde{Y}$ computed by any of these methods satisfies

$$|\tilde{Y} - p(X)| \leq O(d\mathbf{u})(|p_0| + |p_1||X| + |p_2||X|^2 + \cdots + |p_d||X|^d).$$

All OK if p and X only contain nonnegative values, but in all cases in which there is cancellation this could be troublesome (an example later).

Approximating with polynomials

How stable is matrix function evaluation by diagonalization?

Numerically, even if diagonal values are computed "perfectly" $|f(\lambda_i) - \tilde{f}(\lambda_i)| < \varepsilon$, we only have

$$\|f(A) - \tilde{f}(A)\| = \|V(f(\Lambda) - \tilde{f}(\Lambda))V^{-1}\| \leq \kappa(V)\varepsilon,$$

so you may expect trouble if A is non-diagonalizable (again!) or close to it.

One needs to study these approximation properties directly "at the matrix level".

Convergence of Taylor series

Theorem [Higham book Thm. 4.7]

Suppose $f = \sum_{k=0}^{\infty} f_k (x-\alpha)^k$, with $f_k = \frac{f^{(k)}(\alpha)}{k!}$, is a Taylor series with convergence radius r.
Then,

$$\lim_{d\to\infty} \sum_{k=0}^{d} f_k (A - \alpha I)^k = f(A)$$

for each A whose eigenvalues satisfy $|\lambda_i - \alpha| < r$.

Proof:

- Taylor polynomials $p_d(x) = \sum_{k=0}^{d} f_k (x-\alpha)^k$ converge (uniformly) to $f(x)$ when $|x-\alpha| < r$
- $r^{-1} = \limsup (f_k)^{1/k}$.
- $p_d^{(k)}(x)$ is the Taylor polynomial of $f^{(k)}$ (of degree $d-k$), and it has the same radius of convergence.
- If $f_n \to f$ 'with enough derivatives', $f_n(A) \to f(A)$.

The problem with Taylor

Taylor series do not solve every problem satisfactorily.

Example: exponential of a 2×2 matrix.

$$A = \begin{bmatrix} 0 & \alpha \\ -\alpha & 0 \end{bmatrix}, \quad \exp(A) = \begin{bmatrix} \cos\alpha & \sin\alpha \\ -\sin\alpha & \cos\alpha \end{bmatrix}.$$

For $\alpha = 30$, even summing a lot of terms gives poor precision, because the intermediate terms of the series grow a lot (the "hump phenomenon") with respect to the final result: cancellation.

In the scalar case, we can solve the problem by switching to the alternative formula $\exp(A) = (\exp(-A))^{-1}$, but not in the matrix case.

Padé approximations

Variant: Padé approximations, i.e., rational approximations.

Padé approximant (at $x = 0$)

For almost every f analytic at 0 and for every choice of degrees $\deg p, \deg q$, one can find a rational function $\frac{p(x)}{q(x)}$ such that

$$f(x) - \frac{p(x)}{q(x)} = \mathcal{O}(x^{\deg p + \deg q + 1}).$$

i.e., "matches first $\deg p + \deg q$ terms of the MacLaurin series". (Count degrees of freedom to get a hint of why it works.)

Proof: series expansion of $f(x)q(x) = p(x)$ gives a linear system.

For many functions, Padé approximants converge faster than Taylor series.

We will examine them for specific functions, e.g. the exponential.

Parlett recurrence

When Jordan is unstable, use Schur.

Can one compute matrix functions using the Schur form of A?

Example

$$A = \begin{bmatrix} t_{11} & t_{12} \\ 0 & t_{22} \end{bmatrix}, \quad f(A) = \begin{bmatrix} s_{11} & s_{12} \\ 0 & s_{22} \end{bmatrix}.$$

Clearly, $s_{11} = f(t_{11})$, $s_{22} = f(t_{22})$.

Trick: expanding $Af(A) = f(A)A$, one gets an equation for s_{12}:

$$t_{11}s_{12} + t_{12}s_{22} = s_{11}t_{12} + s_{12}t_{22} \implies s_{12} = t_{12}\frac{s_{11} - s_{22}}{t_{11} - t_{22}}.$$

(If $t_{11} = t_{22}$, the equation is not solvable and we already know that the finite difference becomes a derivative).

Parlett recurrence — II

The same idea works for larger blocks (provided we compute things in the correct order):

$$A = \begin{bmatrix} t_{11} & t_{12} & t_{13} \\ & t_{22} & t_{23} \\ & & t_{33} \end{bmatrix}, \quad f(A) = \begin{bmatrix} s_{11} & s_{12} & s_{13} \\ & s_{22} & s_{23} \\ & & s_{33} \end{bmatrix},$$

$$t_{11}{\color{red}s_{13}} + t_{12}s_{23} + t_{13}s_{33} = s_{11}t_{13} + s_{12}t_{23} + {\color{red}s_{13}}t_{33}.$$

Very similar to the algorithm we used to solve Sylvester equations. In some sense, we are solving the (singular) Sylvester equation $AX - XA = 0$ for $X = f(A)$, after setting specific elements on its diagonal.

The same idea works blockwise: the quotients become Sylvester equations.

Parlett recurrence — III

Algorithm (Schur–Parlett method)

1. Compute Schur form $A = QTQ^*$;
2. Partition T into blocks with 'well-separated eigenvalues';
3. Compute $f(T_{ii})$ (e.g., with a Taylor series centered in the average of the cluster);
4. Use recurrences to compute off-diagonal blocks of $f(T)$;
5. Return $f(A) = Qf(T)Q^*$.

Tries to get 'best of both worlds': uses Taylor expansion when the eigenvalues are close, recurrences when they are distant.

Matlab's `funm` does this (for selected functions, or when the user provides derivatives).

Parlett recurrence and block diagonalization

The Parlett recurrence is related to block diagonalization.

Consider the case of 2 blocks for simplicity. T can be block-diagonalized via

$$W^{-1}TW = \begin{bmatrix} I & -X \\ 0 & I \end{bmatrix} \begin{bmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{bmatrix} \begin{bmatrix} I & X \\ 0 & I \end{bmatrix} = \begin{bmatrix} T_{11} & \\ & T_{22} \end{bmatrix}$$

where X solves $T_{11}X - XT_{22} + T_{12} = 0$ (Sylvester equation). Then

$$f(T) = W \begin{bmatrix} f(T_{11}) & \\ & f(T_{22}) \end{bmatrix} W^{-1} = \begin{bmatrix} f(T_{11}) & Xf(T_{22}) - f(T_{11})X \\ & f(T_{22}) \end{bmatrix}.$$

(Note indeed that $S = Xf(T_{22}) - f(T_{11})X$ solves the Sylvester equation appearing in the Parlett recurrence.)

So both methods solve a Sylvester equation with operator $Z \mapsto T_{11}Z - ZT_{22}$.