# Matrix Functions: Theory and Algorithms

Nick Higham
Department of Mathematics
University of Manchester

higham@ma.man.ac.uk
http://www.ma.man.ac.uk/~higham/

Includes joint work with Philip Davies

# OUTLINE

- *Definitions of $f(A)$*
- Applications
- Algorithms for particular $f$
- Schur–Parlett algorithm for general $f$
- Computing $f(A)b$

# Defining by Substitution

Want to define $f : \mathbb{C}^{n\times n} \to \mathbb{C}^{n\times n}$, but *not elementwise*.
Given $f(t)$, can define $f(A)$ by substituting $A$ for $t$:

$$f(t) = \frac{1+t^2}{1-t} \quad \Rightarrow \quad f(A) = (I-A)^{-1}(I+A^2).$$

$$\log(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} - \frac{x^4}{4} + \cdots, \quad |x| < 1$$

$$\Rightarrow \quad \log(I+A) = A - \frac{A^2}{2} + \frac{A^3}{3} - \frac{A^4}{4} + \cdots, \quad \rho(A) < 1.$$

Works for $f$

- a polynomial,
- a rational,
- or with a convergent power series.

# Multiplicity of Definitions

*There have been proposed in the literature since 1880*
*eight distinct definitions*
*of a matric function,*
*by Weyr, Sylvester and Buchheim,*
*Giorgi, Cartan, Fantappiè, Cipolla,*
*Schwerdtfeger and Richter.*

— R. F. Rinehart,
*The Equivalence of Definitions of a Matric Function,*
*Amer. Math. Monthly (1955)*

# Cauchy Integral Theorem

**Definition 1**

$$f(A) = \frac{1}{2\pi i} \int_{\Gamma} f(z)(zI - A)^{-1}\, dz,$$

*where $f$ is analytic inside a closed contour $\Gamma$ which encloses $\lambda(A)$.*

# Jordan Canonical Form

$$Z^{-1}AZ = J = \mathrm{diag}(J_1, J_2, \ldots, J_p), \qquad J_k = \begin{bmatrix} \lambda_k & 1 & & \\ & \lambda_k & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_k \end{bmatrix}$$

**Definition 2**

$$f(A) = Zf(J)Z^{-1} = Z\mathrm{diag}(f(J_k))Z^{-1},$$

$$f(J_k) = \begin{bmatrix} f(\lambda_k) & f'(\lambda_k) & \ldots & \dfrac{f^{(k-1)}(\lambda_k)}{(k-1)!} \\ & f(\lambda_k) & \ddots & \vdots \\ & & \ddots & f'(\lambda_k) \\ & & & f(\lambda_k) \end{bmatrix}.$$

# Interpolation

**Definition 3 (Sylvester, 1883; Buchheim, 1886)** *Distinct e'vals $\lambda_1, \ldots, \lambda_s$, $n_i =$ geometric mult. of $\lambda_i$. Then $f(A) = r(A)$, where $r$ is unique Hermite interpolating poly of degree less than $\sum_{i=1}^{s} n_i$ satisfying interpolation conditions*

$$r^{(j)}(\lambda_i) = f^{(j)}(\lambda_i), \qquad j = 0 : n_i - 1, \quad i = 1 : s.$$

- Poly $r$ depends on $A$.
- This def. preserves functional relations $G(f_1, \ldots, f_p) = 0$, where $G$ is a polynomial. E.g. $\sin^2(A) + \cos^2(A) = I$. But of course $e^{A+B} \neq e^A e^B$.

# Non-Primary Functions

Horn & Johnson call these defs *primary matrix functions*.

But not all possible functions captured when multiple eigenvalues. E.g.,

$$A = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}, \qquad X = \begin{bmatrix} i & 0 \\ 0 & -i \end{bmatrix}, \quad Y = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}.$$

$X$ and $Y$ are square roots of $A$ but are not polynomials in $A$. However, $A = \mathrm{givens}(\pi)$ and $Y = \mathrm{givens}(\pi/2)$ is a natural square root.

- Virtually all existing theory and methods are for primary functions.
- Non-primary functions sometimes needed when tracking $f(A(t))$ when eigenvalues of $A(t)$ coalesce.

# Textbook References

[1] F. R. Gantmacher. *The Theory of Matrices*, volume one. Chelsea, New York, 1959.

[2] Gene H. Golub and Charles F. Van Loan. *Matrix Computations*. Johns Hopkins University Press, Baltimore, MD, USA, third edition, 1996.

[3] Roger A. Horn and Charles R. Johnson. *Topics in Matrix Analysis*. Cambridge University Press, 1991.

[4] Peter Lancaster and Miron Tismenetsky. *The Theory of Matrices*. Academic Press, London, second edition, 1985.

# OUTLINE

Definitions of $f(A)$

- ***Applications***

Algorithms for particular $f$

Schur–Parlett algorithm for general $f$

Computing $f(A)b$

# Application: Differential equations

*Nuclear magnetic resonance*: Solomon equations

$$dM/dt = -RM, \qquad M(0) = I,$$

where $M(t) =$ matrix of intensities and $R =$ symmetric relaxation matrix. NMR workers need to solve both forward and inverse problems.

*Exponential time differencing* for stiff systems (Cox & Matthews, J. Comp. Phys., 2002)

$$y' = Ay + F(y,t).$$

Methods based on exact integration of linear part—require one accurate evaluation of $\exp(hA)$ per integration.

# Application: Control theory

Convert *continuous-time system*

$$\frac{dx}{dt} = Ax(t) + Bu(t)$$

to *discrete-time state-space system*

$$x_{k+1} = Fx_k + Gu_k,$$

where $F = e^{A\tau}$ and $\tau$ is sampling period.
(E.g., MATLAB Control System Toolbox, `c2d`, `d2c`.)

# OUTLINE

Definitions of $f(A)$

Applications

► *Algorithms for particular* $f$

Schur–Parlett algorithm for general $f$

Computing $f(A)b$

# Classic MATLAB

```
      < M A T L A B >
    Version of 01/10/84

 HELP is available

<>
help
 Type HELP followed by
 INTRO   (To get started)
 NEWS    (recent revisions)
 ABS   ANS   ATAN  BASE  CHAR  CHOL  CHOP  CLEA  COND  CONJ  COS
 DET   DIAG  DIAR  DISP  EDIT  EIG   ELSE  END   EPS   EXEC  EXIT
 EXP   EYE   FILE  FLOP  FLPS  FOR   FUN   HESS  HILB  IF    IMAG
 INV   KRON  LINE  LOAD  LOG   LONG  LU    MACR  MAGI  NORM  ONES
 ORTH  PINV  PLOT  POLY  PRIN  PROD  QR    RAND  RANK  RCON  RAT
 REAL  RETU  RREF  ROOT  ROUN  SAVE  SCHU  SHOR  SEMI  SIN   SIZE
 SQRT  STOP  SUM   SVD   TRIL  TRIU  USER  WHAT  WHIL  WHO   WHY
 < > ( ) = . , ; \ / ' + - * :
```

# Classic MATLAB

```
<>
help fun

FUN For matrix arguments  X , the  functions  SIN,  COS,  ATAN,
    SQRT,  LOG,  EXP and X**p are computed using eigenvalues  D
    and eigenvectors  V .  If  <V,D> =  EIG(X)   then   f(X)  =
    V*f(D)/V  .   This method may give inaccurate results if  V
    is badly conditioned.  Some idea of  the  accuracy  can  be
    obtained by comparing  X**1  with  X .
    For vector arguments,  the  function  is  applied  to  each
    component.
```

*The availability of* [FUN] *in early versions of MATLAB*
*quite possibly contributed to*
*the system's technical and commercial success.*

— Cleve Moler (2003)

# Setup

- General nonsymmetric $A$
- Factorization of $A$ feasible
- May not want full accuracy

- Many applications.
- Methods for very large, sparse $A$, often require solution of smaller, dense subproblems.

# Matrix Exponential

- Cleve Moler and Charles Van Loan.
  *Nineteen dubious ways to compute the exponential of a matrix, twenty-five years later*, SIAM Rev., 45 (2003).

  ▷ 355 citations on Science Citation Index.

- Scaling and squaring (SS) method for $X \approx e^A$ (Ward, 1977; Moler & Van Loan, 1978).

  1. $A \leftarrow A/2^k$ so $\|A\|_\infty \le 1/2$
  2. $r(A) =$ [6/6] Padé approximant to $e^A$
  3. $X = r(A)^{2^k}$

Used by MATLAB's `expm`.

# Alternative SS Algorithm for $e^A$

Suggested by Najfeld & Havel (1995): exploit

$$\tau(A) = A\coth(A) = A(e^{2A}+I)(e^{2A}-I)^{-1}$$

$$= I + \cfrac{A^2}{3I + \cfrac{A^2}{5I + \cfrac{A^2}{7I + \cdots}}}.$$

1. $B = A/2^{k+1}$ so $\|A^2\|_\infty/2^{2k+2} \le 1.152$
2. $r(B) =$ [8/8] Padé approximant to $\tau(B)$.
3. $X = \left[(r(B)+B)(r(B)-B)^{-1}\right]^{2^k}$

- Claimed to require fewer flops than original SS alg.

# Principal Log and $p$th Root

Let $A \in \mathbb{C}^{n\times n}$ have no eigenvalues on $\mathbb{R}^-$.

## Log

$X = \log A$ denotes unique $X$ such that

1. $e^X = A$.
2. $-\pi < \operatorname{Im}(\lambda(X)) < \pi$.

## $p$th root

For integer $p > 0$, $X = A^{1/p}$ is unique $X$ such that

1. $X^p = A$.
2. $-\pi/p < \arg(\lambda(X)) < \pi/p$.

# Briggs' Log Method (1617)

$$\log(ab) = \log a + \log b \quad \Rightarrow \quad \log a = 2 \log a^{1/2}.$$

Use repeatedly:

$$\log a = 2^k \log a^{1/2^k}.$$

Write $a^{1/2^k} = 1 + x$ and note $\log(1+x) \approx x$. Briggs worked to base 10 and used

$$\log_{10} a \approx 2^k \cdot \log_{10} e \cdot (a^{1/2^k} - 1).$$

# Briggs' Log Method (1617)

$$\log(ab) = \log a + \log b \quad \Rightarrow \quad \log a = 2\log a^{1/2}.$$

Use repeatedly:

$$\log a = 2^k \log a^{1/2^k}.$$

Write $a^{1/2^k} = 1 + x$ and note $\log(1+x) \approx x$. Briggs worked to base 10 and used

$$\log_{10} a \approx 2^k \cdot \log_{10} e \cdot (a^{1/2^k} - 1).$$

*Briggs must be viewed as one of the great figures in numerical analysis.*

— Herman H. Goldstine, A History of Numerical Analysis (1977)

# Briggs' Log Method (1617)

$$\log(ab) = \log a + \log b \quad \Rightarrow \quad \log a = 2\log a^{1/2}.$$

Use repeatedly:

$$\log a = 2^k \log a^{1/2^k}.$$

Write $a^{1/2^k} = 1 + x$ and note $\log(1+x) \approx x$. Briggs worked to base 10 and used

$$\log_{10} a \approx 2^k \cdot \log_{10} e \cdot (a^{1/2^k} - 1).$$

Can we generalize to matrices:

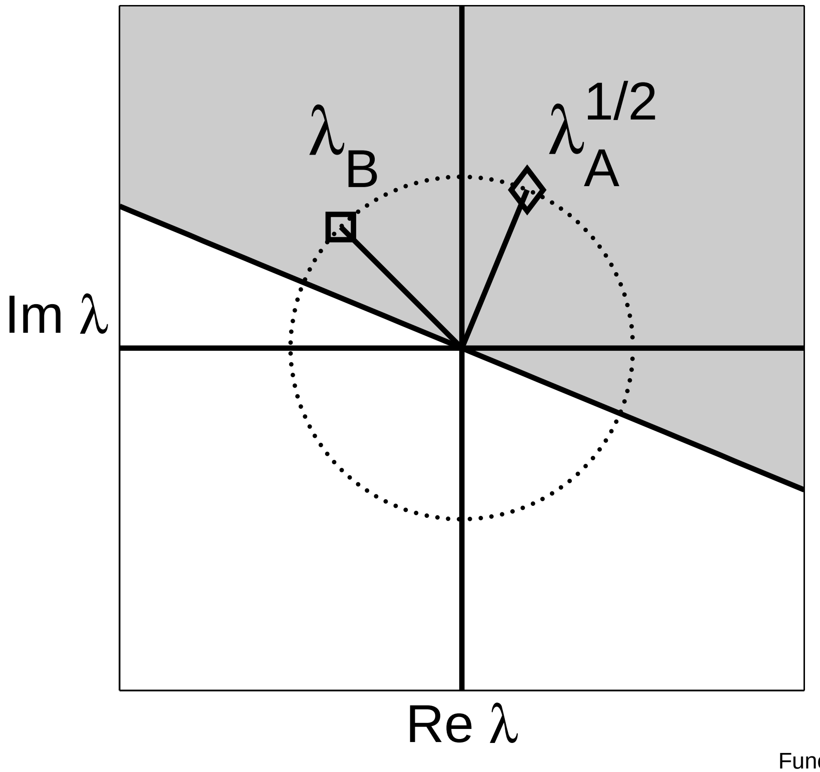
$\log A = 2^k \log A^{1/2^k}$?

# Splitting Lemma

**Lemma 0 (Cheng, H, Kenney & Laub, 2001)** *Suppose* $A = BC$ *has no eigenvalues on* $\mathbb{R}^-$ *and*

1. $BC = CB$.
2. *Every eigenvalue of* $B$ *(or* $C$*) lies in the open halfplane of the corresponding eigenvalue of* $A^{1/2}$.

*Then* $\log A = \log B + \log C$.



# Matrix Logarithm

Use the Briggs idea:

$$\log A = 2^k \log A^{1/2^k}.$$

Kenney & Laub's (1989) inverse scaling and squaring method:

- Bring $A$ close to $I$ by repeated square roots.
- Approximate $\log A^{1/2^k}$ using an $[m/m]$ Padé approximant $r_m(x) \approx \log(1-x)$.
- Rescale to find $\log A$.

# Alg of Cheng, H, Kenney & Laub (2001)

★ Transformation-free: uses only matrix mult, LU, inv.

# Alg of Cheng, H, Kenney & Laub (2001)

- Transformation-free: uses only matrix mult, LU, inv.
- Sq. roots by product form of Denman–Beavers iteration:

$$
\begin{aligned}
M_{k+1} &= \frac{1}{2}\Big[I + \frac{1}{2}(M_k + M_k^{-1})\Big], & M_0 &= A, \\
Y_{k+1} &= Y_k(I + M_k^{-1})/2, & Y_0 &= A,
\end{aligned}
$$

where $M_k \to I$ and $Y_k \to A^{1/2}$.

# Alg of Cheng, H, Kenney & Laub (2001)

- ★ Transformation-free: uses only matrix mult, LU, inv.
- ★ Sq. roots by product form of Denman–Beavers iteration:

$$M_{k+1} = \frac{1}{2}\Big[I + \frac{1}{2}(M_k + M_k^{-1})\Big], \qquad M_0 = A,$$
$$Y_{k+1} = Y_k(I + M_k^{-1})/2, \qquad Y_0 = A,$$

  where $M_k \to I$ and $Y_k \to A^{1/2}$.
- ★ Aims for a specified accuracy.

# Alg of Cheng, H, Kenney & Laub (2001)

- Transformation-free: uses only matrix mult, LU, inv.
- Sq. roots by product form of Denman–Beavers iteration:

$$M_{k+1} = \frac{1}{2}\Big[I + \frac{1}{2}(M_k + M_k^{-1})\Big], \qquad M_0 = A,$$
$$Y_{k+1} = Y_k(I + M_k^{-1})/2, \qquad Y_0 = A,$$

  where $M_k \to I$ and $Y_k \to A^{1/2}$.
- Aims for a specified accuracy.
- Padé degree $m$ chosen using K & L's (1989) bound:

$$\|r_m(X) - \log(I - X)\| \le |r_m(\|X\|) - \log(1 - \|X\|)|.$$

# Alg of Cheng, H, Kenney & Laub (2001)

- Transformation-free: uses only matrix mult, LU, inv.
- Sq. roots by product form of Denman–Beavers iteration:

$$M_{k+1} = \frac{1}{2}\Big[I + \frac{1}{2}(M_k + M_k^{-1})\Big], \qquad M_0 = A,$$
$$Y_{k+1} = Y_k(I + M_k^{-1})/2, \qquad Y_0 = A,$$

  where $M_k \to I$ and $Y_k \to A^{1/2}$.
- Aims for a specified accuracy.
- Padé degree $m$ chosen using K & L's (1989) bound:

$$\|r_m(X) - \log(I - X)\| \le |r_m(\|X\|) - \log(1 - \|X\|)|.$$

- $r_m$ evaluated using partial fraction expansion $r_m(x) = \sum_{j=1}^{m} \frac{\alpha_j^{(m)} x}{1+\beta_j^{(m)} x}$: fast and accurate (H, 2001).

# Matrix $p$th Root

Square root: Björck & Hammarling (1983). Compute Schur decomp. $A = QTQ^*$ and then solve $R^2 = T$ by

$$r_{ii} = \sqrt{t_{ii}}, \quad r_{ij} = \frac{t_{ij} - \sum_{k=i+1}^{j-1} t_{ij} t_{kj}}{t_{ii} + t_{jj}}.$$

Extended to $p$th roots by Smith (2003)—much more complicated recurrence.

These algs

- Have essentially optimal numerical stability.
- Generalize to real Schur decomp.

# Matrix Cosine

**Algorithm 0 (Serbin & Blalock, 1980)** *Given* $A \in \mathbb{R}^{n\times n}$ *and parameter* $\alpha > 0$ *this alg approximates* $\cos(A)$.

Choose $m$ such that $2^{-m}\|A\| \approx \alpha$.
$C_0 =$ Taylor or Pade approximation to $\cos(A/2^m)$.
for $i = 0\colon m-1$
$\quad C_{i+1} = 2C_i^2 - I$
end

- Choice of $m$ (i.e., $\alpha$)?
- Which approximation?
- Effect of rounding errors?

# Alg of H & Smith (2002)

- Initial argument reduction and balancing to reduce norm.
- [8/8] Padé approximation proved fully accurate in IEEE double if $\|A\|_\infty \le 1$. More economical than Taylor series.
- "Schoolboy" evaluation of $r_8(A)$.
- Total cost: $(4 + \lceil \log_2(\|A\|_\infty) \rceil)M + D$.
- Error analysis give bound containing terms $(4.1)^m$ and norms of intermediate $C_i$.

# Numerical Stability

- Is $\|\widehat{f} - f\|$ consistent with condition of problem?
- Is $\widehat{f} = f(A + E)$ with $E$ “small’, i.e., is residual $f^{-1}(\widehat{f}) - A$ “small’?

Unclear for all algs discussed except “yes” for $A^{1/p}$.

★ Currently lack characterizations of when an $f(A)$ problem is ill conditioned for nonnormal $A$.

# OUTLINE

Definitions of $f(A)$

Applications

Algorithms for particular $f$

► *Schur–Parlett algorithm for general* $f$

Computing $f(A)b$

# Similarity Transformations

Can use the formula

$$A = XBX^{-1} \quad \Rightarrow \quad f(A) = Xf(B)X^{-1},$$

provided $f(B)$ is easily computable.
E.g. $B = \mathrm{diag}(\lambda_i)$ if $A$ diagonalizable.

**Problem**: any error $\Delta B$ in $f(B)$ magnified by up to $\kappa(X) = \|X\|\|X^{-1}\| \geq 1$.

Prefer to work with *unitary* $X$: thus can use

- eigendecomposition (diagonal $B$) when $A$ is normal ($AA^* = A^*A$),
- Schur decomposition (triangular $B$) in general.

# Example: Eigendecomposition

```
function F = funm_ev(A,fun)
[V,D] = eig(A);
F = V * diag(feval(fun,diag(D))) / V;

>> A = [3 -1; 1 1]; X = funm_ev(A,@sqrt)
X =
  1.7678e+000 -3.5355e-001
  3.5355e-001  1.0607e+000

>> norm(A-X^2)                    % cond(V) = 9.4e7
ans =
  9.9519e-009

>> Y = sqrtm(A); norm(A-Y^2)
ans =
  6.4855e-016
```

# Parlett's Recurrence

Schur decomposition $A = QTQ^*$ reduces problem to $F = f(T)$, $T$ upper triangular.

$f_{ii} = f(t_{ii})$ is immediate.

Parlett (1976): from $FT = TF$ obtain recurrence

$$f_{ij} = t_{ij}\frac{f_{ii} - f_{jj}}{t_{ii} - t_{jj}} + \sum_{k=i+1}^{j-1} \frac{f_{ik}t_{kj} - t_{ik}f_{kj}}{t_{ii} - t_{jj}}.$$

Used in MATLAB's `funm`.

# Parlett's Recurrence

Schur decomposition $A = QTQ^*$ reduces problem to $F = f(T)$, $T$ upper triangular.

$f_{ii} = f(t_{ii})$ is immediate.

Parlett (1976): from $FT = TF$ obtain recurrence

$$f_{ij} = t_{ij} \frac{f_{ii} - f_{jj}}{t_{ii} - t_{jj}} + \sum_{k=i+1}^{j-1} \frac{f_{ik} t_{kj} - t_{ik} f_{kj}}{t_{ii} - t_{jj}}.$$

Used in MATLAB's `funm`.

*Fails* when $T$ has repeated eigenvalues.

# Parlett vs. Björck & Hammarling

Parlett recurrence is not "optimal", as clear from sq. root case: $x_{12}$ obtained from

$$\text{Parlett}: \quad \frac{a_{12}(\sqrt{a_{11}} - \sqrt{a_{22}})}{a_{11} - a_{22}} = \frac{a_{12}}{\sqrt{a_{11}} + \sqrt{a_{22}}} \quad : \text{B \& H.}$$

# Schur–Parlett Algorithm

H & Davies (2002):

- Compute Schur decomposition $A = QTQ^*$.
- Re-order $T$ to block triangular form in which eigenvalues within a block are “close” and those of separate blocks are “well separated”.
- Evaluate $F_{ii} = f(T_{ii})$.
- Solve the Sylvester equations

$$T_{ii}F_{ij} - F_{ij}T_{jj} = F_{ii}T_{ij} - T_{ij}F_{jj} + \sum_{k=i+1}^{j-1} (F_{ik}T_{kj} - T_{ik}F_{kj}).$$

- Undo the unitary transformations.

# Function of Atomic Block

Assume $f$ has Taylor series with $\infty$ radius of cgce and derivatives available.

For diagonal blocks $T$ use

$$T = \sigma I + M, \quad \sigma = \mathrm{trace}(T)/n : \qquad f(T) = \sum_{k=0}^{\infty} \frac{f^{(k)}(\sigma)}{k!} M^k.$$

- Truncate series based on strict error bound, *not* using size of terms. NB: for $n = 2$,

$$M = \begin{bmatrix} \epsilon & \alpha \\ 0 & -\epsilon \end{bmatrix}$$

$$\Rightarrow \quad M^{2k} = \begin{bmatrix} \epsilon^{2k} & 0 \\ 0 & \epsilon^{2k} \end{bmatrix}, \quad M^{2k+1} = \begin{bmatrix} \epsilon^{2k+1} & \alpha\epsilon^{2k} \\ 0 & -\epsilon^{2k+1} \end{bmatrix}.$$

# Features of Algorithm

- Costs $O(n^3)$ flops, or up to $n^4/3$ flops if large blocks needed (close, repeated eigenvalues).
- Needs derivatives if blocks size $> 1$: price to pay for treating general $f$ and nonnormal $A$.
- Best general $f(A)$ alg. Benchmark for comparing other $f(A)$ algs—general and specific.
- The basis of a new `funm` for next MATLAB release.

# OUTLINE

Definitions of $f(A)$

Applications

Algorithms for particular $f$

Schur–Parlett algorithm for general $f$

► *Computing* $f(A)b$

# $\log(A)\, b$

Apply quadrature rule $\int_0^1 f(t)\,dt \approx \sum_{k=1}^m c_k f(t_k)$ to (Wouk, 1965)

$$\log A = \int_0^1 (A-I)\big[t(A-I)+I\big]^{-1}\,dt.$$

Combine with Hessenberg reduction $A = QHQ^T$ to get

$$(\log A)\, b \approx Q \sum_{k=1}^m c_k \big[t_k(H-I)+I\big]^{-1} d, \qquad d = Q^T(A-I)b,$$

Costs $(10/3)n^3 + 2mn^2$ flops.

- When $\|I-A\| < 1$ can use $m$-point Gauss-Legendre $\equiv$ Padé approximation! Choose $m$ using (Kenney & Laub, 2001)

$$\|r_{mm}(X) - \log(I+X)\| \le |r_{mm}(-\|X\|) - \log(1-\|X\|)|.$$

- When $\|I-A\| > 1$ use adaptive quadrature.

# $A^\alpha b$

$$\frac{dy}{dt} = \alpha(A - I)[t(A - I) + I]^{-1}y, \qquad y(0) = b$$

has unique solution $y(t) = [t(A - I) + I]^\alpha b \Rightarrow y(1) = A^\alpha b$.
Used by Allen, Baglama & Boyd (2000) for $\alpha = 1/2$, spd $A$.

**Example** using MATLAB's `ode45`.
$A =$ `gallery('parter',64)`, $b =$ `randn(64,1)`.

| $f(A)$ | tol | Succ. steps | Fail. atts | $f$ evals | Rel. err |
|---|---|---|---|---|---|
| $A^{-1/2}$ | 1e-3 | 12 | 0 | 73 | 3.5e-8 |
| | 1e-6 | 14 | 0 | 85 | 6.0e-9 |
| | 1e-9 | 40 | 0 | 241 | 7.7e-12 |
| $A^{2/5}$ | 1e-3 | 15 | 0 | 79 | 2.8e-8 |
| | 1e-6 | 16 | 0 | 91 | 2.4e-9 |
| | 1e-9 | 54 | 0 | 325 | 1.8e-12 |

# Interpolation

If $A$ has distinct eigenvalues $\lambda_j$, Lagrange interp poly:

$$f(A)b = \sum_{j=0}^{n} f_j \ell_j(A)b, \qquad \ell_j(x) = \frac{\prod_{k=0,\ k\neq j}^{n} (x-\lambda_k)}{\prod_{k=0,\ k\neq j}^{n} (\lambda_j - \lambda_k)}.$$

Cost: $O(n^4)$ flops.
For any $A$, Newton divided difference form:

$$f(A)b = \sum_{i=0}^{n} c_i \prod_{j=0}^{i-1} (A - \lambda_j I)b, \qquad c_i = \text{(confluent) div. diffs.}$$

Requires derivatives of $f$. Cost: $O(n^3)$ flops.

# Cauchy Integral Theorem

$$y = \frac{1}{2\pi i} \int_{\Gamma} f(z)(zI - A)^{-1} b \, dz =: \int_{\Gamma} g(z) \, dz.$$

Take circle

$$\Gamma\colon\; z - \alpha = \beta e^{i\theta}, \quad 0 \le \theta \le 2\pi.$$

Apply repeated trapezium rule:

$$\int_{\Gamma} g(z) \; dz = \int_0^{2\pi} (z(\theta) - \alpha) g(z(\theta)) \, d\theta \approx \frac{2\pi i}{n} \sum_{k=0}^{n-1} (z_k - \alpha) g(z_k),$$

where $z_k - \alpha = \beta e^{2\pi k i/n}$.

Use Hessenberg reduction, as before.

# Euler-Maclaurin Error Bound

$h(x)$ period $2\pi$, in $C^{2k+1}(-\infty,\infty)$, $|h^{(2k+1)}(x)| \le M$:

$$\left|\int_0^{2\pi} h(x)\, dx - T_n(f)\right| \le \frac{4\pi M\, \zeta(2k+1)}{n^{2k+1}}.$$

- $h^{(2k+1)}(x)$ proportional to $\beta^{2k+2}$ = radius of circle.
- $h^{(2k+1)}(x)$ contains powers of resolvent $(z(\theta)I - A)^{-1}$. Bad if contour close to some $\lambda_i$ or $A$ highly nonnormal.
- $h^{(2k+1)}(x)$ contains derivatives of $f$ on contour.

Conclude : restricted to matrices

- not too nonnormal,
- $\lambda_i$ can be enclosed in circle of small radius not close to singularity of derivs of $f$.

# Future Work

- Theory and algorithms for non-primary functions, perhaps linked to an $f(A(t))$ application.
- Better understanding of conditioning of $f(A)$.
- Exploiting structure, e.g. $A \in$ matrix automorphism group (H, Mackey, Mackey & Tisseur, 2003).

http://www.ma.man.ac.uk/~higham/