Methods for $A^{1/2}$

Note Title 2023-04-18

Recap: Schur-Parlett

1) $A = QUQ^*$ to reduce to triangular U
2) Compute diagonal of $S = f(U)$ $s_{ii} = f(u_{ii})$
3) Looking at entry (i,j) of the identity $US = SU$, we obtain formulas $s_{ij} = \dfrac{\cdots}{u_{ii} - u_{jj}}$ which we can use to compute entries s_{ij} one by one
4) $f(A) = QSQ^*$

For $f(x) = x^{1/2}$, variant to avoid the denominator: use $S^2 = U$ instead of $SU = US$

$$u_{ij} = s_{ii}s_{ij} + s_{i,i+1}s_{i+1,j} + s_{i,i+2}s_{i+2,j} + \cdots + s_{ij}s_{jj} \qquad i < j$$

$$s_{ij} = \frac{u_{ij} - s_{i,i+1}s_{i+1,j} - \cdots - s_{i,j-1}s_{j-1,j}}{s_{ii} + s_{jj}}$$

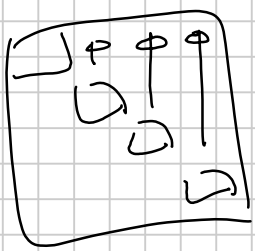
s entries below and to the left of s_{ij}

denominator

$s_{ii} + s_{jj} = u_{ii}^{1/2} + u_{jj}^{1/2}$ cannot be 0: $u_{ii}, u_{jj} \in \mathrm{RHP} \Rightarrow u_{ii} + u_{jj} \in \mathrm{RHP}$

Moreover, $\dfrac{1}{s_{ii} + s_{jj}} = f[u_{ii}, u_{jj}] = \dfrac{u_{ii}^{1/2} - u_{jj}^{1/2}}{u_{ii} - u_{jj}}$ is large only when $f(A)$ is ill-cond.

(Matlab implementation)



```
function B = sqrtm_schurparlett(A)
[Q, U] = schur(A, 'complex');
n = size(A, 1);
S = zeros(n, n);
for j = 1:n
    S(j,j) = sqrt(U(j,j));
    for i = j-1:-1:1
        num = U(i,j) - S(i,i+1:j-1)*S(i+1:j-1,j);
        S(i,j) = num / (S(i,i)+S(j,j));
    end
end
B = Q*S*Q';
if isreal(A)
    B = real(B);
end
```

$\frac{\|\tilde{B}^2 - A\|}{\|A\|} = O(u)$ $\Leftrightarrow$ the computed $\tilde{B}$ is the exact square root of $A + \delta_A$ $\frac{\|\delta_A\|}{\|A\|} = O(u)$

i.e., the method is backward stable.

This method is unfortunately not backward stable, but we can prove that

$$\frac{\|\tilde{B}^2 - A\|}{\|\tilde{B}\|^2} = O(u)$$

(Note that $\|A\| = \|B^2\| = \|B \cdot B\| \leq \|B\| \cdot \|B\|$, and there can be major cancellation for non-normal matrices)

Theorem: suppose $\tilde{S} = f(U)$ is computed in floating point with machine precision u, then

$$\tilde{S}^2 = U + \delta_U \qquad |\delta_U| \leq |S|^2 \, O(n) u$$

$$|\delta U_{ij}| \leq \sum_{k=i}^{j} |S_{ik}| \cdot |S_{kj}|$$, elementwise abs. value and inequalities.

From this theorem, with a little more work we obtain a norm bound. $\|\tilde{B}^2 - A\| \le O(u)\,\|\tilde{B}\|^2$.

Proof: Note first that in the RHS we can freely replace S with $\tilde{S}$, because $S - \tilde{S} = O(u)$

$$\tilde{S}_{ij} = \left(U_{ij} \ominus \tilde{S}_{i,i+1} \odot \tilde{S}_{i+1,j} \ominus \cdots \ominus \tilde{S}_{i,j-1} \odot \tilde{S}_{j-1,j} \right) \oslash \left(\tilde{S}_{ii} \oplus \tilde{S}_{jj} \right)$$

Replace $a \circledast b$ with $(a \circledast b)(1+\varepsilon_k)$ $|\varepsilon_k| \le u$
+ ignore higher-order terms

$$\left| U_{ij} - \tilde{S}_{i,i+1}\tilde{S}_{i+1,j} - \cdots - \tilde{S}_{i,j-1}\tilde{S}_{j-1,j} - \tilde{S}_{ij}(\tilde{S}_{ii} + \tilde{S}_{jj}) \right|$$

$$= |\text{error terms}| \le u\left(|U_{ij}| + |\tilde{S}_{i,i+1}\tilde{S}_{i+1,j}| + \cdots + |\tilde{S}_{i,j-1}\tilde{S}_{j-1,j}| + |\tilde{S}_{ij}|\left(|\tilde{S}_{ii}| + |\tilde{S}_{jj}|\right) \right)$$

e.g. with only one term:

$$\tilde{S}_{ij} = \left(U_{ij} \ominus \tilde{S}_{i,i+1}\tilde{S}_{i+1,j} \right) \oslash \left(\tilde{S}_{ii} \oplus \tilde{S}_{jj} \right)$$

$$= \left(U_{ij} - \tilde{S}_{i,i+1}\tilde{S}_{i+1,j} \right)(1+\varepsilon_1)(1+\varepsilon_3) \Big/ \left(\tilde{S}_{ii} + \tilde{S}_{jj} \right)(1+\varepsilon_2)$$

$$U_{ij}(1+\varepsilon_1)(1+\varepsilon_3) - \left(\tilde{S}_{i,i+1}\tilde{S}_{i+1,j} \right)(1+\varepsilon_1)(1+\varepsilon_3) - \tilde{S}_{ij}\left(\tilde{S}_{ii} + \tilde{S}_{jj} \right)(1+\varepsilon_2) = 0$$

$$U_{ij} - \tilde{S}_{i,i+1}\tilde{S}_{i+1,j} - \tilde{S}_{ij}\tilde{S}_{ii} - \tilde{S}_{ij}\tilde{S}_{jj} = -U_{ij}(\varepsilon_1 + \varepsilon_3) + \left(\tilde{S}_{i,i+1}\tilde{S}_{i+1,j} \right)(\varepsilon_1 + \varepsilon_3)$$
$$+ \tilde{S}_{ij}\left(\tilde{S}_{ii} + \tilde{S}_{jj} \right)\varepsilon_2 + O(u^2)$$

$$|U - \tilde{S}^2|_{ij} \le 2u\left(|U_{ij}| + |\tilde{S}_{i,i+1}\tilde{S}_{i+1,j}| + |\tilde{S}_{ij}\tilde{S}_{ii}| + |\tilde{S}_{ij}\tilde{S}_{jj}| \right)$$

$$\left(|U| + |\tilde{S}|^2 \right)_{ij}$$

$$|U-\tilde{S}^2|_{ij} \le \left(|U|_{ij} + |\tilde{S}|^2_{ij}\right) O(n)u$$

$$|U|_{ij} = |s_{ii}s_{ij} + s_{i,i+1}s_{i+1,j} + \dots + s_{ij}s_{jj}|$$

$$\le |S|_{ii}|S|_{ij} + |S|_{i,i+1}|S|_{i+1,j} + \dots + |s_{ij}||s_{jj}| = |S|^2_{ij}$$

then $|U-\tilde{S}^2|_{ij} \le |S|^2_{ij} O(n)u$

Matlab's sqrtm( ) uses a similar algorithm:
a recursive block version of this: split U in blocks

$$U = \left[\begin{array}{c|c} U_{11} & U_{12} \\ \hline 0 & U_{22} \end{array}\right]$$ of equal (± 1) size

then compute $S_{11} = (U_{11})^{\frac{1}{2}}$ $S_{22} = (U_{22})^{\frac{1}{2}}$ recursively and

S_{12} by solving a Sylvester equation that comes from

$$\left[\begin{array}{c|c} S_{11} & S_{12} \\ \hline 0 & S_{22} \end{array}\right]\left[\begin{array}{c|c} S_{11} & S_{12} \\ \hline 0 & S_{22} \end{array}\right] = \left[\begin{array}{c|c} U_{11} & U_{12} \\ \hline 0 & U_{22} \end{array}\right]$$

$S_{11}S_{12} + S_{12}S_{22} = U_{12}$ a Sylvester equation.

Another method: matrix iterations.

$B = A^{\frac{1}{2}}$ solves $A - B^2 = 0$, i.e., it is a zero of ~~$f(X) = A - X^2$~~ $f(X) = X^2 - A$.

Multivariate Newton method: $F: \mathbb{C}^m \to \mathbb{C}^m$

$$\begin{cases} x_0 \in \mathbb{C}^m \text{ given} \\ x_{k+1} = x_k - \left(\text{Jac}F_{x_k}\right)^{-1} F(x_k) \end{cases}$$

$x_{k+1} = x_k - h$ where h solves

$\uparrow$ $m\times m$ $\quad$ $\uparrow$ $m\times 1$ $\quad$ $(\mathrm{Jac}\, F_{X_k})\,\mathrm{vec}(H) = F(X_k)$

We can apply this method working on the "vectorized form" of f.

$m = n^2 \qquad F(x) = \mathrm{vec}\left((\mathrm{unvec}(x))^2 - A \right)$

$\mathrm{Jac}\, F_X$ is the Kronecker form of the Fréchet derivative $\quad L_{f,X}[E] = XE + EX$

So re-seen as a matrix iteration the Newton method becomes

$$\begin{cases} X_0 \in \mathbb{C}^{n\times n} \text{ given} \\ X_{k+1} = X_k - H \quad \text{where } H \text{ solves } X_k H + H X_k = X_k^2 - A \end{cases}$$

Multivariate Newton method: Starting from given X_0:

For $k = 0, 1, 2, \dots$

1. Solve $X_k H + H X_k = X_k^2 - A$
2. $X_{k+1} = X_k - H$

We expect this to converge quadratically, as Newton methods do.

Problem: to solve the Sylvester equation we need a Schur form of X_k at each step.

However, there is a simpler way to avoid the Sylvester solution

Lemma: If you start from X_0 s.t. $AX_0 = X_0 A$ (e.g. $X_0 = \alpha \cdot I$, for $\alpha > 0$, or $X_0 = \alpha A$), then

1. At each step, X_k and A commute
2. At each step, $H = (2X_k)^{-1}(X_k^2 - A)$

Proof: combined induction.

For $k=0$, 1. is obvious

Once proved 1. for a certain k, we can prove 2.:

$$X_k H + H X_k = X_k (2X_k)^{-1}(X_k^2 - A) + (2X_k)^{-1}(X_k^2 - A) X_k$$

$$= \left(X_k (2X_k)^{-1} + (2X_k)^{-1} X_k \right)(X_k^2 - A)$$

$$= \left(\tfrac{1}{2} I + \tfrac{1}{2} I \right)(X_k^2 - A) = X_k^2 - A$$

H, being a simple expression in X_k and A, commutes, too.

$\Rightarrow$ $X_{k+1} = X_k - H$ commutes with A, too.

This gives a cheaper version of the method

$$\begin{pmatrix} \text{Schur} & \sim 30n^3 \\ \text{Inversion} & \sim 2n^3 \end{pmatrix}$$

So we can modify the Newton iteration to

$$X_{k+1} = X_k - (2X_k)^{-1}(X_k^2 - A) = X_k - \tfrac{1}{2}\left(X_k^{-1} X_k^2 - X_k^{-1} A \right)$$

$$= \frac{1}{2}\left(X_k + X_k^{-1} A \right)$$

Modified Newton method:

Given X_0 that commutes with A:

for $k = 0, 1, 2, \ldots$

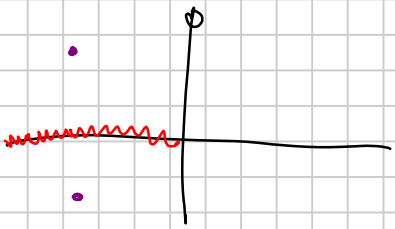
$$X_{k+1} = \frac{1}{2}\left(X_k + X_k^{-1} A \right)$$

end

MN computes the same iterates as the "true Newton" method, at least in exact arithmetic.

These iterates converge quadratically to a solution of $X^2 - A = 0$.

Theorem: Assume A has no real non-positive eigenvalues, and that the iteration is started from $X_0 = \alpha I$ or $X_0 = \alpha A$ with $\alpha > 0$.
Then, the iterations converge to the principal square root $A^{\frac{1}{2}}$



Proof: set $Z_k := A^{-\frac{1}{2}} X_k$. Then,

$$Z_{k+1} = A^{-\frac{1}{2}} X_{k+1} = A^{-\frac{1}{2}} \frac{1}{2}\left(X_k + X_k^{-1} A\right) = \frac{1}{2}\left(A^{-\frac{1}{2}} X_k + X_k^{-1} A^{\frac{1}{2}}\right)$$

$$= \frac{1}{2}\left(Z_k + Z_k^{-1}\right)$$

This is the Newton iteration for the matrix sign!

Z_k converges quadratically to $\operatorname{sign}(Z_0)$

$$Z_0 = \begin{cases} A^{-\frac{1}{2}} \cdot \alpha I = \alpha \cdot A^{-\frac{1}{2}} \\ A^{-\frac{1}{2}} \alpha A = \alpha A^{\frac{1}{2}} \end{cases}$$

have all eigenvalues in RHP since $\Lambda(A^{\frac{1}{2}}) \subset$ RHP

Hence, $\operatorname{sign}(Z_0) = I$

So $Z_k \to I$, and $X_k \to A^{\frac{1}{2}}$ where $k \to \infty$.
$A^{-\frac{1}{2}} X_k$

However, in practice their behavior is very different! ⚠
We will see it in the following lecture

```
rng(0);
M = randn(10); M = M*M';

X = eye(size(M));
Y = eye(size(M));
for k = 1:15
    X = X - lyap(X, X, M-X^2); % True Newton
    Y = 1/2 * (Y + Y\M);
    TNres(k) = norm(X^2-M) / norm(M);
    MNres(k) = norm(Y^2-M) / norm(M);
    TNcom(k) = norm(M*X-X*M) / (norm(M)*norm(X));
    MNcom(k) = norm(M*Y-Y*M) / (norm(M)*norm(Y));
end
[(1:15)' TNres' MNres' TNcom' MNcom']
```

```
>> sqrt_comparison
ans =
   1.0000e+00   1.2716e+01   1.2716e+01   5.0496e-17   5.0496e-17
   2.0000e+00   2.9472e+00   2.9472e+00   2.7535e-16   1.6357e-16
   3.0000e+00   5.5013e-01   5.5013e-01   8.5127e-16   5.6933e-16
   4.0000e+00   4.8810e-02   4.8810e-02   4.6606e-16   3.8247e-15
   5.0000e+00   5.6788e-04   5.6788e-04   9.7668e-17   2.9243e-14
   6.0000e+00   8.0577e-08   8.0577e-08   7.0494e-17   2.2360e-13
   7.0000e+00   1.6465e-15   1.8219e-12   8.2882e-17   1.7098e-12
   8.0000e+00   1.0450e-16   1.3934e-11   9.4525e-17   1.3078e-11
   9.0000e+00   1.1462e-16   1.0658e-10   5.7112e-17   1.0003e-10
   1.0000e+01   8.5601e-17   8.1522e-10   1.1203e-16   7.6519e-10
   1.1000e+01   9.6578e-17   6.2358e-09   8.2755e-17   5.8532e-09
   1.2000e+01   1.0385e-16   4.7700e-08   6.9193e-17   4.4773e-08
   1.3000e+01   9.9863e-17   3.6488e-07   1.0014e-16   3.4249e-07
   1.4000e+01   1.0058e-16   2.7911e-06   7.0488e-17   2.6198e-06
   1.5000e+01   1.0279e-16   2.1350e-05   5.8032e-17   2.0040e-05
```

TN works well, but MN stops converging at residual $\approx 10^{-12}$!