Defining a Residual for Matrix Functions

James Munch

Math 789R

October 10, 2014
Matrix Exponential and its applications

Matrix Exponential

\[ e^A = \sum_{k=0}^{\infty} \frac{A^k}{k!} \]

Things we know about \( e^A \):

- The matrix exponential has many applications in complex network analysis.
- Specific entries of \( e^A \) (e.g. \( A \) is the adjacency or graph laplacian of a network) or products with vectors give information about the corresponding network, such as different centrality measures.
- The power series representation of the matrix exponential is not useful in practice for computing the matrix explicitly.
- \( e^A \) loses sparsity patterns of \( A \). We want methods that, in general, do not compute \( e^A \) in its entirety when not needed.
- Can compute \( e^A \) using iterative methods, approximating by finite polynomials with Krylov subspace methods (Arnoldi, Lanczos, SaI)
1. Defining Residuals

2. Krylov Subspace Methods

3. Relaxation Methods

4. Community Detection

5. Bibliography
Let $A \in \mathbb{R}^{n \times n}, \ v \in \mathbb{R}^n$. We want to compute $y = e^{-A}v$ using some iterative scheme. So, we need a way to determine how good of an approximation $y_k$ is to the true solution $y$.

Instead, consider $y$ as the solution to the IVP at time $t = 1$:

$$\begin{cases}
y'(t) = -Ay \\
y(0) = v
\end{cases}$$

<table>
<thead>
<tr>
<th>Explicit Formulation</th>
<th>$x = A^{-1}b$</th>
<th>$y = e^{-A}v$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Residual</td>
<td>$Ax_k - b$</td>
<td>$-Ay_k(t) - y'(t)$</td>
</tr>
<tr>
<td>Error</td>
<td>$A\epsilon_k = r_k$</td>
<td>$r_k = \epsilon_k(t)' + A\epsilon_k(t), \ \epsilon(0) = 0$</td>
</tr>
</tbody>
</table>
1. Defining Residuals
2. Krylov Subspace Methods
3. Relaxation Methods
4. Community Detection
5. Bibliography
Arnoldi/Lanczos Process

We begin by decomposing $A$ using an Arnoldi/Lanczos process. Let $A \in \mathbb{R}^{n \times n}$, $V_k \in \mathbb{R}^{n \times k}$, $H_k \in \mathbb{R}^{k \times k}$ upper Hessenberg,

$$AV_k = V_k H_k + h_{k+1,k} v_k e_k^T$$

$$V_k = [v_1 \ v_2 \ \ldots \ v_k], \quad \text{span} \{v_i\} = \mathcal{K}_k (v, Av, A^2 v, \ldots, A^k v)$$

$$y_k(t) = V_k e^{-tH_k} (\beta e_1), \quad \beta = \| v \|$$

$$= V_k u_k(t), \quad u_k(t) = e^{-tH_k} (\beta e_1)$$

Where $u_k$ is the solution to the “projected IVP”:

$$\begin{align*}
    u_k'(t) &= -H_k u_k(t) \\
    u_k(0) &= \beta e_1
\end{align*}$$
Arnoldi Residual

Theorem (Botchev)

Let $y_k(t) \approx e^{-tA}v$ be the Krylov approximate solution. Then:

$$r_k = \beta h_{k+1,k} e_k^T e^{-tH_k} e_1 v_{k+1}$$

$$\|r_k\| = |h_{k+1,k} [u_k(t)]_k|$$
Useful properties of Ritz-Galerkin Residual

- $y_k$ satisfies the initial condition by construction
- These residuals give us orthogonal search directions
- Error can be approximated without Krylov restarting: (Eshof/Hochbruch 2006)

\[ \|y_{k+m} - y_k\| = \|e_k\| \approx \|\hat{e}_k\| = \|u_{k+m} - u_k\| \]
Convergence analysis

Figure 6.4. Convergence of the conventional Arnoldi method with two existing stopping criteria and Krylov-Richardson with the residual-based stopping criterion for tolerance \( \text{toler} = 10^{-5} \). Left: stagnation-based criterion, Arnoldi stops too early (201 matevecs, 2.6 s CPU time, error 1.0e−03). Right: generalised residual criterion, Arnoldi stops too late (487 matevecs, 139 s CPU time, error 4.9e−08). Parameters of the Krylov-Richardson run for both plots: 434 matevecs, 11 s CPU time, error 2.2e−06. The CPU measurements (on a 3GHz Linux PC) are made in Matlab and thus are only an indication of the actual performance.

Figure 6.5. Convergence plots of the Arnoldi/Lanczos and the new Krylov-Richardson methods, mesh 102 x 102, Pe = 100. Left: restart every 15 steps, right: Saf strategy with GMRES. The peaks in the residual plots on the left correspond to the restarts.

Figure: Comparison of error using new residual to previous standards
1. Defining Residuals

2. Krylov Subspace Methods

3. Relaxation Methods

4. Community Detection

5. Bibliography
Relaxed Richardson Iteration

In general for a linear system, we write the Richardson iteration as:

$$x_{k+1} = x_k + \omega r_k$$

(1)

where \( \omega \) is chosen so that the fixed point iteration is a contraction:

$$\rho(I - \omega A) < 1.$$ 

Note: this is equivalent to gradient descent for normal equations for solving least squares problems.
Preconditioned Richardson for the Matrix Exponential

For a linear system $Ax = b$ with $Ae_k = r_k$ we can look at the fixed point iteration for $M \approx A$:

$$x_{k+1} = x_k + M^{-1}r_k \approx x_k + e_k$$  \hspace{1cm} (2)

We choose $M$ so that $M$ is both a good approximation of $A$ while being easier to “invert”.

Substituting these iterations for our analogous concepts for the matrix exponential, we get:

\[
\begin{align*}
\chi_{k+1} &= \chi_k + \epsilon_k \\
\epsilon'_{k+1} &= -M\epsilon_k + r_k, \quad \epsilon_k(0) = 0 \\
r_k &= -Ax_k - x'_k,
\end{align*}
\]
Convergence Analysis

We will take the Laplace transform of our iteration and find $R_s$ such that $e_{k+1} = R_s e_k$ where $e_k = \mathbb{X} - X_k$ is the error in the Laplace space and $\mathbb{X}$ is the true solution.

$$\mathcal{L}(\chi) := X, \quad \mathcal{L}(\epsilon) := E$$

$$sE_k - \epsilon_k(0) = -ME_k - AX_k - sX_k + \chi_k(0)$$
$$sE_k + ME_k = -sX_k - AX_k + \chi_k(0)$$
$$(sl + M)E_k = -(sl + A)X_k + \chi_k(0)$$

$$E_k = -(sl + M)^{-1}(sl + A)X_k + (sl + M)^{-1}\chi_k(0)$$

$$sl + M := M_s, \quad sl + A := A_s$$
$$\mathbb{X} - X_{k+1} = \mathbb{X} - M_s^{-1}A_s\mathbb{X} + M_s^{-1}\chi(0) - X_k + M_s^{-1}A_sX_k - M_s^{-1}\chi_k(0)$$

$$e_{k+1} = (I - M_s^{-1}A_s)e_k$$

$$R_s = I - M_s^{-1}A_s$$
Figure: Spectral radius of $R_s$ as a function of frequency space parameter.

Note that for proper choices of $M$, we get $\rho(R_s) < 1 \forall s$ but that $\lim_{s \to \infty} \rho(R_s) = 0$. We can perform similar analysis on different splitting schemes and see that using the residual as so defined acts as a relaxation operator on our method.
1. Defining Residuals
2. Krylov Subspace Methods
3. Relaxation Methods
4. Community Detection
5. Bibliography
Community Detection: current work

Let \( L = I - D^{-1/2}AD^{-1/2} \) be the doubly stochastic graph Laplacian. Then \( y(t) = e^{-tL}v \) gives us a notion for the spread of information over a graph given an initial distribution of information given by \( v \).

Hypothesis: Given \( v_i = \pi \) (node \( i \) is in a particular community) then there exists \( \hat{t}, \gamma \) such that \( C \approx \{ i \mid y(\hat{t})_i > \gamma \} \). That is to say, starting with some initial distribution of information, \( e^{-tA} \) tells us how information has traveled over the graph over a given interval of time. We would expect information to spread quickly within communities and take longer to flow outside of a given community. At some time, we should be able to detect this phenomena by a large gap in the values of \( y(t) \).

Algorithm:
- Establish relaxation method (Jacobi, SOR, Gauss-Seidel)
- Compute \( R_k = [r_k(t_0) \ r_k(t_1) \ldots \ r_k(t_n)] \), \( r_k(t) = -Ly - y' \)
- Update \( y(t)_i \) where \( ||R_k(i,:)|| \) is large.
1. Defining Residuals
2. Krylov Subspace Methods
3. Relaxation Methods
4. Community Detection
5. Bibliography
• Residual, restarting and Richardson iteration for the matrix exponential, revised. Botchev, 2006
Thank you!