

Quadrature-Based Arnoldi Restarts for Matrix Functions with Randomized Sketching

Jingyu Liu

February 9, 2026

1 Introduction

This note considers the computation of $f(\mathbf{A})\mathbf{b}$ where $\mathbf{A} \in \mathbb{C}^{N \times N}$ is a matrix, \mathbf{b} is a vector in \mathbb{C}^N and f is a suitable matrix function. Typically, \mathbf{A} is non-Hermitian, large but sparse.

2 Quadrature-Based Arnoldi Restarts

A commonly used method is the Krylov subspace method. Suppose we have the Arnoldi decomposition

$$\mathbf{A}\mathbf{V}_m = \mathbf{V}_{m+1}\underline{\mathbf{H}}_m = \mathbf{V}_m\mathbf{H}_m + \mathbf{v}_{m+1}h_{m+1,m}\mathbf{e}_m^\top \quad (2.1)$$

for the Krylov subspace $\mathcal{K}_m(\mathbf{A}; \mathbf{b})$, where $\mathbf{V}_m \in \mathbb{C}^{N \times m}$ is orthonormal and $\underline{\mathbf{H}}_m \in \mathbb{C}^{(m+1) \times m}$ is upper-Hessenberg. The starting vector \mathbf{v}_1 is chosen to be $\mathbf{b}/\|\mathbf{b}\|$. If we define $\mathbf{H}_m = \mathbf{V}_m^*\mathbf{A}\mathbf{V}_m$, then the Arnoldi approximation of $f(\mathbf{A})\mathbf{b}$ is given by

$$\mathbf{f}_m = \mathbf{V}_m f(\mathbf{H}_m) \mathbf{V}_m^* \mathbf{b} = \|\mathbf{b}\| \mathbf{V}_m f(\mathbf{H}_m) \mathbf{e}_1. \quad (2.2)$$

Our goal is to find a way to restart for the Krylov method. We first give a formula for the approximation error. Suppose $\Omega \subset \mathbb{C}$ is a region and $f: \Omega \rightarrow \mathbb{C}$ is analytic with the integral representation

$$f(z) = \int_\Gamma \frac{g(t)}{t - z} dz, \quad z \in \Omega \quad (2.3)$$

where $\Gamma \subset \mathbb{C} \setminus \Omega$ is a closed contour and $g: \Gamma \rightarrow \mathbb{C}$ is a known suitable function. We have the following theorem.

Theorem 2.1 (Error of the Arnoldi approximation, [2]). *Suppose f has an integral representation as in (2.3). Denote \mathbf{f}_m the Arnoldi approximation (2.2) to $f(\mathbf{A})\mathbf{b}$. Let $\text{spec}(\mathbf{H}_m) = \{\theta_1, \dots, \theta_m\} \subset \Omega$, $\phi_m(t) = (t - \theta_1) \cdots (t - \theta_m)$ and $\gamma_m = \prod_{j=1}^m h_{j+1,j}$. Then, the error of \mathbf{f}_m can be expressed as*

$$f(\mathbf{A})\mathbf{b} - \mathbf{f}_m = \gamma_m \|\mathbf{b}\| \int_\Gamma \frac{g(t)}{\phi_m(t)} (t\mathbf{I} - \mathbf{A})^{-1} dt \mathbf{v}_{m+1} =: \text{err}_m(\mathbf{A})\mathbf{v}_{m+1}, \quad (2.4)$$

provided the integral exists. Therefore, the error function is given by

$$\text{err}_m(z) = \gamma_m \|\mathbf{b}\| \int_\Gamma \frac{g(t)}{\phi_m(t)} (t - z)^{-1} dt. \quad (2.5)$$

The proof of this theorem is based on the following three results.

Lemma 2.2. *Let \mathbf{f}_m be the Arnoldi approximation (2.2) to $f(\mathbf{A})\mathbf{b}$. Then, $\mathbf{f}_m = p_{m-1}(\mathbf{A})\mathbf{b}$ where p_{m-1} is the unique polynomial of degree at most $m - 1$ that interpolates f at the eigenvalues of \mathbf{H}_m .*

Lemma 2.3. *Suppose f has an integral representation as in (2.3). The interpolating polynomial p_{m-1} with interpolation points $\theta_1, \dots, \theta_m \subset \Omega$ is given as*

$$p_{m-1}(z) = \int_{\Gamma} \left(1 - \frac{\phi_m(z)}{\phi_m(t)}\right) \frac{g(t)}{t - z} dt, \quad (2.6)$$

where $\phi_m(z) = (z - \theta_1) \cdots (z - \theta_m)$, provided the integral in (2.6) exists.

Lemma 2.4. *Let $p(z) = c_0 + c_1 z + \cdots + c_m z^m$ be the unique polynomial of degree exact m . Then,*

$$p(\mathbf{A})\mathbf{b} = \|\mathbf{b}\|(\mathbf{V}_m p(\mathbf{H}_m)\mathbf{e}_1 + \mathbf{v}_{m+1} \gamma_m c_m).$$

Using \mathbf{v}_{m+1} as the new starting vector, suppose we have a new Arnoldi decomposition

$$\mathbf{A}\mathbf{V}_m^+ = \mathbf{V}_{m+1}^+ \underline{\mathbf{H}}_m^+$$

with $\mathbf{v}_1^+ = \mathbf{v}_{m+1}$. The correction term can be computed through

$$\text{err}_m(\mathbf{A})\mathbf{v}_{m+1} = \mathbf{V}_m^+ \text{err}_m(\mathbf{H}_m^+) \mathbf{V}_m^{+,*} \mathbf{v}_{m+1} = \mathbf{V}_m^+ \text{err}_m(\mathbf{H}_m^+) \mathbf{e}_1.$$

This gives a general framework for restarting the Arnoldi method for matrix functions, as stated in algorithm 1.

Algorithm 1 Restarted FOM approximation for $f(\mathbf{A})\mathbf{b}$

Input: Matrix \mathbf{A} , vector \mathbf{b} , function f , subspace dimension m .

Output: $\mathbf{f}_m^{[k]} \approx f(\mathbf{A})\mathbf{b}$.

- 1: Compute the Arnoldi decomposition $\mathbf{A}\mathbf{V}_m^{[0]} = \mathbf{V}_{m+1}^{[0]} \underline{\mathbf{H}}_m^{[0]}$ where $\mathbf{v}_1^{[0]} = \mathbf{b}/\|\mathbf{b}\|$.
 - 2: $\mathbf{f}_m^{[0]} = \|\mathbf{b}\| \mathbf{V}_m^{[0]} f(\mathbf{H}_m^{[0]}) \mathbf{e}_1$.
 - 3: **for** $k = 1, 2, \dots$ **do**
 - 4: Determine the error function $\text{err}_m^{[k-1]}(z)$.
 - 5: Compute the Arnoldi decomposition $\mathbf{A}\mathbf{V}_m^{[k]} = \mathbf{V}_{m+1}^{[k]} \underline{\mathbf{H}}_m^{[k]}$ where $\mathbf{v}_1^{[k]} = \mathbf{v}_{m+1}^{[k-1]}$.
 - 6: Update the approximation $\mathbf{f}_m^{[k]} = \mathbf{f}_m^{[k-1]} + \mathbf{V}_m^{[k]} \text{err}_m^{[k-1]}(\mathbf{H}_m^{[k]}) \mathbf{e}_1$.
 - 7: **end for**
-

We have the following easy-to-verify corollary.

Corollary 2.5 ([2]). *Under the same assumptions as Theorem 2.1 after k ($k \geq 0$) restarts can be expressed as*

$$f(\mathbf{A})\mathbf{b} - \mathbf{f}_m^{[k]} = \gamma_m^{[0]} \cdots \gamma_m^{[k]} \|\mathbf{b}\| \int_{\Gamma} \frac{g(t)}{\phi_m^{[0]}(t) \cdots \phi_m^{[k]}(t)} (t\mathbf{I} - \mathbf{A})^{-1} \mathbf{v}_{m+1}^{[k]} dt =: \text{err}_m^{[k]}(\mathbf{A})\mathbf{v}_{m+1}^{[k]}.$$

Numerically, the computation in terms of err_m refers to a quadrature formula

$$\widehat{\text{err}}_m(z) = \gamma_m \|\mathbf{b}\| \sum_{i=1}^{\ell} w_i \frac{g(t_i)}{\phi_m(t_i)} \frac{1}{t_i - z}.$$

Combing all these together, we have the quadrature-based restart Arnoldi approximation as in algorithm 2. One important feature of this algorithm is that the number of quadrature points ℓ can be adapted during the iterations to achieve a desired accuracy.

Algorithm 2 Quadrature-based restart Arnoldi approximation for $f(\mathbf{A})\mathbf{b}$

Input: Matrix \mathbf{A} , vector \mathbf{b} , function f , subspace dimension m , initial number of quadrature points ℓ , quadrature tolerance τ .

Output: $\mathbf{f}_m^{[k]} \approx f(\mathbf{A})\mathbf{b}$.

- 1: Compute the Arnoldi decomposition $\mathbf{AV}_m^{[0]} = \mathbf{V}_{m+1}^{[0]} \underline{\mathbf{H}}_m^{[0]}$ where $\mathbf{v}_1^{[0]} = \mathbf{b}/\|\mathbf{b}\|$.
 - 2: $\mathbf{f}_m^{[0]} = \|\mathbf{b}\| \mathbf{V}_m^{[0]} f(\underline{\mathbf{H}}_m^{[0]}) \mathbf{e}_1$.
 - 3: $\ell = \lceil \ell/\sqrt{2} \rceil$.
 - 4: **for** $k = 1, 2, \dots$ **do**
 - 5: Determine the error function $\text{err}_m^{[k-1]}(z)$ according to Corollary 2.5.
 - 6: Compute the Arnoldi decomposition $\mathbf{AV}_m^{[k]} = \mathbf{V}_{m+1}^{[k]} \underline{\mathbf{H}}_m^{[k]}$ where $\mathbf{v}_1^{[k]} = \mathbf{v}_{m+1}^{[k-1]}$.
 - 7: Choose quadrature points and weights $\{(t_i, w_i)\}_{i=1}^\ell$ and $\{(\tilde{t}_i, \tilde{w}_i)\}_{i=1}^{\tilde{\ell}}$.
 - 8: Set accurate = false and refined = false.
 - 9: **while** accurate = false **do**
 - 10: Compute $\mathbf{h}_m^{[k]} = \widehat{\text{err}}_m^{[k-1]}(\underline{\mathbf{H}}_m^{[k]}) \mathbf{e}_1$ using $\{(t_i, w_i)\}_{i=1}^\ell$.
 - 11: Compute $\tilde{\mathbf{h}}_m^{[k]} = \widehat{\text{err}}_m^{[k-1]}(\underline{\mathbf{H}}_m^{[k]}) \mathbf{e}_1$ using $\{(\tilde{t}_i, \tilde{w}_i)\}_{i=1}^{\tilde{\ell}}$.
 - 12: **if** $\|\mathbf{h}_m^{[k]} - \tilde{\mathbf{h}}_m^{[k]}\| \leq \tau \|\tilde{\mathbf{h}}_m^{[k]}\|$ **then**
 - 13: accurate = true.
 - 14: **else**
 - 15: refined = true.
 - 16: Update $\tilde{\ell} = \ell$, $\ell = \lceil \sqrt{2}\tilde{\ell} \rceil$.
 - 17: **end if**
 - 18: **end while**
 - 19: Update the approximation $\mathbf{f}_m^{[k]} = \mathbf{f}_m^{[k-1]} + \mathbf{V}_m^{[k]} \mathbf{h}_m^{[k]}$.
 - 20: **if** refined = false **then**
 - 21: Update $\ell = \tilde{\ell}$, $\tilde{\ell} = \lceil \ell/\sqrt{2} \rceil$.
 - 22: **end if**
 - 23: **end for**
-

3 Randomized Sketching for Computing Matrix Functions

Consider the computation of $f(\mathbf{A})\mathbf{b}$. Using the integral definition of matrix functions, we have

$$f(\mathbf{A})\mathbf{b} = \frac{1}{2\pi i} \int_C f(t)(t\mathbf{I} - \mathbf{A})^{-1} \mathbf{b} dt = \frac{1}{2\pi i} \int_C f(t) \mathbf{x}(t) dt,$$

where C is a closed contour enclosing $\text{spec}(\mathbf{A})$ and $\mathbf{x}(t)$ is the solution to the shifted linear system $(t\mathbf{I} - \mathbf{A})\mathbf{x}(t) = \mathbf{b}$.

Using this integral representation, the Arnoldi approximant can be expressed as

$$\mathbf{f}_m = \|\mathbf{b}\| \mathbf{V}_m f(\mathbf{H}_m) \mathbf{e}_1 = \|\mathbf{b}\| \mathbf{V}_m \frac{1}{2\pi i} \int_C f(t)(t\mathbf{I} - \mathbf{H}_m)^{-1} \mathbf{e}_1 dt = \frac{1}{2\pi i} \int_C f(t) \mathbf{x}_m(t) dt,$$

where $\mathbf{x}_m(t) = \|\mathbf{b}\| \mathbf{V}_m(t\mathbf{I} - \mathbf{H}_m)^{-1} \mathbf{e}_1$. In fact, $\mathbf{x}_m(t)$ is the Arnoldi approximation to the shifted linear system $(t\mathbf{I} - \mathbf{A})\mathbf{x}(t) = \mathbf{b}$. To see this, suppose $\mathbf{x}(t)$ is given by $\mathbf{x}_m(t) = \mathbf{V}_m \mathbf{y}_m(t)$ where $\mathbf{y}_m(t)$ is to be determined. Define the residual $\mathbf{r}_m(t) = \mathbf{b} - (t\mathbf{I} - \mathbf{A})\mathbf{x}_m(t)$. The Galerkin condition requires that the residual is orthogonal to $\mathcal{K}_m(\mathbf{A}; \mathbf{b})$, i.e., $\mathbf{V}_m^* \mathbf{r}_m(t) = 0$. This leads to $\mathbf{y}_m(t) = \|\mathbf{b}\| (t\mathbf{I} - \mathbf{H}_m)^{-1} \mathbf{e}_1$.

Suppose now we have a basis (not necessarily orthonormal) $\mathbf{B}_m \in \mathbb{C}^{N \times m}$ for the Krylov subspace $\mathcal{K}_m(\mathbf{A}; \mathbf{b})$. We want to approximate $\mathbf{x}(t)$ by a vector $\mathbf{x}_m(t) = \mathbf{B}_m \mathbf{y}_m(t)$. The Galerkin condition gives

$$\begin{aligned} \mathbf{y}_m(t) &= [(t\mathbf{B}_m^* \mathbf{B}_m - \mathbf{B}_m^* \mathbf{A} \mathbf{B}_m)]^{-1} \mathbf{B}_m^* \mathbf{b} \\ &= [(t\mathbf{I} - (\mathbf{B}_m^* \mathbf{B}_m)^{-1} \mathbf{B}_m^* \mathbf{A} \mathbf{B}_m)]^{-1} (\mathbf{B}_m^* \mathbf{B}_m)^{-1} \mathbf{B}_m^* \mathbf{b} \\ &= [(t\mathbf{I} - \mathbf{B}_m^\dagger \mathbf{A} \mathbf{B}_m)]^{-1} \mathbf{B}_m^\dagger \mathbf{b} \end{aligned}$$

and

$$\begin{aligned} \mathbf{f}_m &= \frac{1}{2\pi i} \int_C f(t) \mathbf{x}_m(t) dt \\ &= \frac{1}{2\pi i} \int_C f(t) \mathbf{B}_m [(t\mathbf{I} - \mathbf{B}_m^\dagger \mathbf{A} \mathbf{B}_m)]^{-1} \mathbf{B}_m^\dagger \mathbf{b} dt \\ &= \mathbf{B}_m f(\mathbf{B}_m^\dagger \mathbf{A} \mathbf{B}_m) \mathbf{B}_m^\dagger \mathbf{b} \end{aligned} \tag{3.1}$$

This is called the FOM approximation. When $\mathbf{B}_m = \mathbf{V}_m$ is orthonormal, it can be directly found that the FOM approximation becomes the Arnoldi approximation.

We use a randomized sketching technique to give another approximation for $\mathbf{x}(t)$. Suppose it is given by $\hat{\mathbf{x}}_m(t) = \mathbf{B}_m \hat{\mathbf{y}}_m(t)$ where $\hat{\mathbf{y}}_m(t)$ is to be determined. Let $\mathbf{S} \in \mathbb{C}^{s \times N}$ ($m \leq s \ll N$) be a sketching matrix such that for some $\varepsilon \in [0, 1]$ for all vectors \mathbf{v} in $\mathcal{K}_{m+1}(\mathbf{A}; \mathbf{b})$,

$$(1 - \varepsilon) \|\mathbf{v}\|^2 \leq \|\mathbf{S}\mathbf{v}\|^2 \leq (1 + \varepsilon) \|\mathbf{v}\|^2. \tag{3.2}$$

Unlike the Arnoldi approximation, instead of requiring the residual $\hat{\mathbf{r}}_m(t) = \mathbf{b} - (t\mathbf{I} - \mathbf{A})\hat{\mathbf{x}}_m(t)$ to be orthogonal to the Krylov subspace $\mathcal{K}_m(\mathbf{A}; \mathbf{b})$, we require the sketched residual $\mathbf{S}\hat{\mathbf{r}}_m(t)$ to be orthogonal to the sketched Krylov subspace $\mathbf{S}\mathcal{K}_m(\mathbf{A}; \mathbf{b})$, i.e., $(\mathbf{S}\mathbf{B}_m)^* \mathbf{S}\hat{\mathbf{r}}_m(t) = 0$. This gives

$$\hat{\mathbf{y}}_m(t) = [(\mathbf{S}\mathbf{B}_m)^* (t\mathbf{S}\mathbf{B}_m - \mathbf{S}\mathbf{A}\mathbf{B}_m)]^{-1} (\mathbf{S}\mathbf{B}_m)^* (\mathbf{S}\mathbf{b})$$

and

$$\hat{\mathbf{f}}_m = \frac{1}{2\pi i} \int_C \hat{\mathbf{x}}_m(t) dt = \mathbf{B}_m \frac{1}{2\pi i} \int_C f(t) [(\mathbf{S}\mathbf{B}_m)^* (t\mathbf{S}\mathbf{B}_m - \mathbf{S}\mathbf{A}\mathbf{B}_m)]^{-1} (\mathbf{S}\mathbf{B}_m)^* (\mathbf{S}\mathbf{b}) dt. \tag{3.3}$$

This is called the sketched FOM approximant of $f(\mathbf{A})\mathbf{b}$.

Suppose $\mathbf{S}\mathbf{B}_m$ has full column rank (and this is always true), then

$$\begin{aligned} [(\mathbf{S}\mathbf{B}_m)^*(t\mathbf{S}\mathbf{B}_m - \mathbf{S}\mathbf{A}\mathbf{B}_m)]^{-1} &= [t(\mathbf{S}\mathbf{B}_m)^*(\mathbf{S}\mathbf{B}_m) - (\mathbf{S}\mathbf{B}_m)^*(\mathbf{S}\mathbf{A}\mathbf{B}_m)]^{-1} \\ &= \left\{ t\mathbf{I} - [(\mathbf{S}\mathbf{B}_m)^*(\mathbf{S}\mathbf{B}_m)]^{-1}(\mathbf{S}\mathbf{B}_m)^*(\mathbf{S}\mathbf{A}\mathbf{B}_m) \right\}^{-1} [(\mathbf{S}\mathbf{B}_m)^*(\mathbf{S}\mathbf{B}_m)]^{-1} \\ &= \left\{ t\mathbf{I} - (\mathbf{S}\mathbf{B}_m)^\dagger(\mathbf{S}\mathbf{A}\mathbf{B}_m) \right\}^{-1} [(\mathbf{S}\mathbf{B}_m)^*(\mathbf{S}\mathbf{B}_m)]^{-1} \end{aligned}$$

Now,

$$\begin{aligned} \hat{\mathbf{f}}_m &= \mathbf{B}_m \frac{1}{2\pi i} \int_C f(t) \left\{ t\mathbf{I} - (\mathbf{S}\mathbf{B}_m)^\dagger(\mathbf{S}\mathbf{A}\mathbf{B}_m) \right\}^{-1} dt \\ &\quad [(\mathbf{S}\mathbf{B}_m)^*(\mathbf{S}\mathbf{B}_m)]^{-1}(\mathbf{S}\mathbf{B}_m)^*(\mathbf{S}\mathbf{b}) \\ &= \mathbf{B}_m f \left((\mathbf{S}\mathbf{B}_m)^\dagger(\mathbf{S}\mathbf{A}\mathbf{B}_m) \right) (\mathbf{S}\mathbf{B}_m)^\dagger(\mathbf{S}\mathbf{b}) \end{aligned} \tag{3.4}$$

Moreover, if $\mathbf{S}\mathbf{B}_m$ is orthonormal, then (3.4) reduces to

$$\hat{\mathbf{f}}_m = \mathbf{B}_m f \left((\mathbf{S}\mathbf{B}_m)^*(\mathbf{S}\mathbf{A}\mathbf{B}_m) \right) (\mathbf{S}\mathbf{B}_m)^*(\mathbf{S}\mathbf{b}). \tag{3.5}$$

When $\mathbf{S}\mathbf{B}_m$ is not orthonormal, we can use the thin QR decomposition $\mathbf{S}\mathbf{B}_m = \mathbf{Q}_m \mathbf{R}_m$ where $\mathbf{Q}_m \in \mathbb{C}^{s \times m}$ is orthonormal and $\mathbf{R}_m \in \mathbb{C}^{m \times m}$ is upper-triangular. Substituting this into (3.4), we have

$$\hat{\mathbf{f}}_m = \mathbf{B}_m f \left(\mathbf{R}_m^{-1} \mathbf{Q}_m^*(\mathbf{S}\mathbf{A}\mathbf{B}_m) \right) \mathbf{R}_m^{-1} \mathbf{Q}_m^*(\mathbf{S}\mathbf{b}). \tag{3.6}$$

JINGYU: this expression is a bit different from [3]. This is because I adjust the position of the inverse matrix and the current version is more easy to understand and analyze.

The above discussions are summarized in algorithm 3.

Algorithm 3 Sketched FOM approximation for $f(\mathbf{A})\mathbf{b}$

Input: Matrix \mathbf{A} , vector \mathbf{b} , function f , subspace dimension m .

Output: $\hat{\mathbf{f}}_m^{[k]} \approx f(\mathbf{A})\mathbf{b}$.

- 1: Draw a sketching matrix $\mathbf{S} \in \mathbb{C}^{s \times N}$.
 - 2: Generate a basis $\mathbf{B}_m \in \mathbb{C}^{N \times m}$ for the Krylov subspace $\mathcal{K}_m(\mathbf{A}; \mathbf{b})$ as well as $\mathbf{S}\mathbf{B}_m$ and $\mathbf{S}\mathbf{A}\mathbf{B}_m$.
 - 3: **if** $\mathbf{S}\mathbf{B}_m$ is orthonormal **then**
 - 4: Compute the sketched FOM approximation $\hat{\mathbf{f}}_m$ using (3.5).
 - 5: **else**
 - 6: Compute the thin QR decomposition $\mathbf{S}\mathbf{B}_m = \mathbf{Q}_m \mathbf{R}_m$.
 - 7: Compute the sketched FOM approximation $\hat{\mathbf{f}}_m$ using (3.6).
 - 8: **end if**
-

We give two strategies to generate the basis \mathbf{B}_m . The first one is to use the truncated Arnoldi process, as illustrated in Algorithm 4. The second one is based on the Gram–Schmidt process

on the sketched vectors. This is illustrated in Algorithm 5. Note that in Algorithm 5, we also compute $\mathbf{S}\mathbf{B}_m$ and \mathbf{SAB}_m during the process. Interestingly, in these two algorithms, we also have an Arnoldi-like decomposition, which will be discussed in the next section.

Algorithm 4 Truncated Arnoldi process

Input: Matrix \mathbf{A} , vector \mathbf{b} , subspace dimension m , truancation parameter t .

Output: Basis $\mathbf{B}_m \in \mathbb{C}^{N \times m}$ for the Krylov subspace $\mathcal{K}_m(\mathbf{A}; \mathbf{b})$.

```

1:  $b_1 = \mathbf{b}/\|\mathbf{b}\|$ .
2: for  $j = 1, 2, \dots, m$  do
3:    $\mathbf{w}_j = \mathbf{Ab}_j$ .
4:   for  $i = \max(1, j - t + 1), \dots, j$  do
5:      $h_{i,j} = \mathbf{b}_i^* \mathbf{w}_j$ .
6:      $\mathbf{w}_j = \mathbf{w}_j - \mathbf{b}_i h_{i,j}$ .
7:   end for
8:    $h_{j+1,j} = \|\mathbf{w}_j\|$ .
9:   if  $h_{j+1,j} = 0$  then
10:    Stop.
11:   end if
12:    $\mathbf{b}_{j+1} = \mathbf{w}_j/h_{j+1,j}$ .
13: end for
14:  $\mathbf{B}_m = [\mathbf{b}_1, \dots, \mathbf{b}_m]$ .

```

Algorithm 5 Sketched Arnoldi process

Input: Matrix \mathbf{A} , vector \mathbf{b} , subspace dimension m , sketching dim s .

Output: Basis $\mathbf{B}_m \in \mathbb{C}^{N \times m}$ for the Krylov subspace $\mathcal{K}_m(\mathbf{A}; \mathbf{b})$.

```

1: Draw a sketching matrix  $\mathbf{S} \in \mathbb{C}^{s \times N}$ .
2:  $\mathbf{w}_0 = \mathbf{b}$ ,  $\mathbf{p}_0 = \mathbf{Sw}_0$ 
3:  $\beta = \|\mathbf{p}_0\|$ .
4:  $\mathbf{q}_1 = \mathbf{p}_0/\beta$ ,  $\mathbf{b}_1 = \mathbf{w}_0/\beta$ 
5: for  $j = 1, 2, \dots, m$  do
6:    $\mathbf{w}_j = \mathbf{Ab}_j$ ,  $\mathbf{p}_j = \mathbf{Sw}_j$ .
7:   for  $i = 1, \dots, j$  do
8:      $h_{i,j} = \mathbf{q}_i^* \mathbf{p}_j$ .
9:      $\mathbf{p}_j = \mathbf{p}_j - \mathbf{q}_i h_{i,j}$ ,  $\mathbf{w}_j = \mathbf{w}_j - \mathbf{b}_i h_{i,j}$ .
10:   end for
11:    $h_{j+1,j} = \|\mathbf{p}_j\|$ .
12:   if  $h_{j+1,j} = 0$  then
13:     Stop.
14:   end if
15:    $\mathbf{q}_{j+1} = \mathbf{p}_j/h_{j+1,j}$ ,  $\mathbf{b}_{j+1} = \mathbf{w}_j/h_{j+1,j}$ .
16: end for
17:  $\mathbf{B}_m = [\mathbf{b}_1, \dots, \mathbf{b}_m]$ ,  $\mathbf{SB}_m = [\mathbf{q}_1, \dots, \mathbf{q}_m]$ ,  $\mathbf{SAB}_m = [\mathbf{p}_1, \dots, \mathbf{p}_m]$ .

```

As for the error analysis of the sketched FOM approximant, we have the following result:

Theorem 3.1 (Error of the sketched FOM approximant). *Suppose \mathbf{S} is a sketching matrix of $\mathcal{K}_{m+1}(\mathbf{A}; \mathbf{b})$ satisfying (3.2) with some $\varepsilon \in [0, 1)$ and that $\mathbf{S}\mathbf{B}_m$ is orthonormal. Let \mathbf{f}_m and $\hat{\mathbf{f}}_m$ denote the FOM and sketched FOM approximants for $f(\mathbf{A})\mathbf{b}$, respectively, as defined in (3.1) and (3.5). Then,*

$$\|\mathbf{f}_m - \hat{\mathbf{f}}_m\| \leq \sqrt{\frac{1+\varepsilon}{1-\varepsilon}} \|\mathbf{b}\| \|f(\mathbf{B}_m^\dagger \mathbf{A} \mathbf{B}_m) - f((\mathbf{S}\mathbf{B}_m)^* \mathbf{S}\mathbf{A}\mathbf{B}_m)\|.$$

4 The Arnoldi-like decomposition

In this section, we give some discussions on the Arnoldi-like decomposition. Suppose we have the following relationship:

$$\mathbf{AB}_m = \mathbf{B}_m \mathbf{H}_m + \mathbf{b}_{m+1} h_{m+1,m} \mathbf{e}_m^\top = \mathbf{B}_{m+1} \underline{\mathbf{H}}_m, \quad (4.1)$$

where $\mathbf{B}_m \in \mathbb{C}^{N \times m}$ is a basis of the Krylov subspace $\mathcal{K}_m(\mathbf{A}; \mathbf{b})$ and \mathbf{H}_m is upper-Hessenberg. Then, we have the following lemma, which is an analogue of Lemma 2.4

Lemma 4.1. *Suppose an Arnoldi-like decomposition (4.1) holds. Let $p(z) = c_0 + c_1 z + \cdots + c_m z^m$ be the unique polynomial of degree exact m . Then,*

$$p(\mathbf{A})\mathbf{b}_1 = \mathbf{B}_m p(\mathbf{H}_m) \mathbf{e}_1 + \mathbf{b}_{m+1} \gamma_m c_m,$$

where $\gamma_m = \prod_{j=1}^m h_{j+1,j}$.

Recall that the FOM approximant (3.1) is $\mathbf{f}_m = \mathbf{B}_m f(\mathbf{G}_m) \mathbf{B}_m^\dagger \mathbf{b}$ where $\mathbf{G}_m = \mathbf{B}_m^\dagger \mathbf{A} \mathbf{B}_m$. It is remarkable that if we require \mathbf{b}_{m+1} to be orthogonal to \mathbf{B}_m , then $\mathbf{G}_m = \mathbf{H}_m$. That is, the matrix \mathbf{H}_m in the Arnoldi-like decomposition is exactly the projection of \mathbf{A} onto the Krylov subspace $\mathcal{K}_m(\mathbf{A}; \mathbf{b})$. In this case, the FOM approximant can be expressed as

$$\mathbf{f}_m = \mathbf{B}_m f(\mathbf{H}_m) \mathbf{B}_m^\dagger \mathbf{b}. \quad (4.2)$$

Suppose p_{m-1} is the unique polynomial of degree at most $m-1$ that interpolates f at the eigenvalues of \mathbf{G}_m , then $f(\mathbf{G}_m) = p_{m-1}(\mathbf{G}_m)$. Assume that $\mathbf{b} = \mathbf{b}_1 \beta$, then by Lemma 4.1, we have

$$\begin{aligned} \mathbf{f}_m &= \mathbf{B}_m f(\mathbf{G}_m) \mathbf{B}_m^\dagger \mathbf{b} = \mathbf{B}_m f(\mathbf{G}_m) \mathbf{e}_1 \beta = \mathbf{B}_m p_{m-1}(\mathbf{G}_m) \mathbf{e}_1 \beta \\ &= \mathbf{B}_m p_{m-1}(\mathbf{H}_m) \mathbf{e}_1 \beta = p_{m-1}(\mathbf{A}) \mathbf{b}_1 \beta = p_{m-1}(\mathbf{A}) \mathbf{b}. \end{aligned}$$

Using this formula, we have an analogue result of Theorem 2.1.

Theorem 4.2 (Error of the FOM when \mathbf{b}_{m+1} is orthogonal to \mathbf{B}_m). *Let (4.1) be a given Arnoldi-like decomposition where \mathbf{B}_m is a basis for the Krylov subspace $\mathcal{K}_m(\mathbf{A}; \mathbf{b})$ and $\mathbf{b} = \mathbf{b}_1 \beta$. Suppose f has an integral representation as in (2.3). Denote $\mathbf{f}_m = \mathbf{B}_m f(\mathbf{G}_m) \mathbf{B}_m^\dagger \mathbf{b}$ the FOM approximation to $f(\mathbf{A})\mathbf{b}$ where $\mathbf{G}_m = \mathbf{B}_m^\dagger \mathbf{A} \mathbf{B}_m$. Let $\text{spec}(\mathbf{G}_m) = \{\theta_1, \dots, \theta_m\} \subset \Omega$, $\phi_m(t) = (t - \theta_1) \cdots (t - \theta_m)$ and $\gamma_m = \prod_{j=1}^m h_{j+1,j}$. If $\mathbf{b}_{m+1} \perp \mathbf{B}_m$, then the error of FOM approximation can be expressed as*

$$f(\mathbf{A})\mathbf{b} - \mathbf{f}_m = \gamma_m \beta \int_{\Gamma} \frac{g(t)}{\phi_m(t)} (t\mathbf{I} - \mathbf{A})^{-1} dt \mathbf{b}_{m+1} =: \text{err}_m(\mathbf{A}) \mathbf{b}_{m+1}. \quad (4.3)$$

provided the integral exists. Therefore, the error function is given by

$$\text{err}_m(z) = \gamma_m \beta \int_{\Gamma} \frac{g(t)}{\phi_m(t)} (t - z)^{-1} dt.$$

Proof. By Lemma 2.3, we have

$$p_{m-1}(z) = \int_{\Gamma} \left(1 - \frac{\phi_m(z)}{\phi_m(t)}\right) \frac{g(t)}{t-z} dt,$$

where $\phi_m(z) = (z - \theta_1) \cdots (z - \theta_m)$. Therefore,

$$f(z) - p_{m-1}(z) = \int_{\Gamma} \frac{\phi_m(z)}{\phi_m(t)} \frac{g(t)}{t-z} dt$$

and

$$f(\mathbf{A})\mathbf{b} - \mathbf{f}_m = \int_{\Gamma} \frac{\phi_m(\mathbf{A})}{\phi_m(t)} (t\mathbf{I} - \mathbf{A})^{-1} g(t) dt \mathbf{b}.$$

Again by Lemma 4.1, we have $\phi_m(\mathbf{A})\mathbf{b} = \phi_m(\mathbf{A})\mathbf{b}_1\beta = \mathbf{B}_m\phi_m(\mathbf{H}_m)\mathbf{e}_1\beta + \mathbf{b}_{m+1}\gamma_m\beta$. But since $\mathbf{b}_{m+1} \perp \mathbf{B}_m$, we have $\mathbf{H}_m = \mathbf{G}_m$ and thus $\phi_m(\mathbf{H}_m) = \mathbf{0}$. Then,

$$f(\mathbf{A})\mathbf{b} - \mathbf{f}_m = \gamma_m\beta \int_{\Gamma} \frac{g(t)}{\phi_m(t)} (t\mathbf{I} - \mathbf{A})^{-1} dt \mathbf{b}_{m+1}.$$

This completes the proof. \square

Corollary 4.3. *Under the same assumptions as Theorem 4.2 after k ($k \geq 0$) restarts can be expressed as*

$$f(\mathbf{A})\mathbf{b} - \mathbf{f}_m^{[k]} = (\gamma_m^{[0]} \cdots \gamma_m^{[k]})(\beta^{[0]} \cdots \beta^{[k]}) \int_{\Gamma} \frac{g(t)}{\phi_m^{[0]}(t) \cdots \phi_m^{[k]}(t)} (t\mathbf{I} - \mathbf{A})^{-1} \mathbf{b}_{m+1}^{[k]} dt =: \text{err}_m^{[k]}(\mathbf{A})\mathbf{b}_{m+1}^{[k]},$$

where we assume $\mathbf{b} = \beta^{[0]}\mathbf{b}_1^{[0]}$, $\mathbf{b}_{m+1}^{[j-1]} = \beta^{[j]}\mathbf{b}_1^{[j]}$ for $1 \leq j \leq k$, and that $\mathbf{b}_{m+1}^{[j]}$ is orthogonal to $\mathbf{B}_m^{[j]}$ for $0 \leq j \leq k$.

The corresponding functions about the above discussions are:

- “funm_quad_fom_last_orth_tarnoldi”
- “funm_quad_fom_last_orth_sarnoldi”

That is, after generating a basis \mathbf{B} using the truncated Arnoldi process (Algorithm 4) or sketched Arnoldi process (Algorithm 5), we can perform an additional step to orthogonalize \mathbf{b}_{m+1} against \mathbf{B}_m and then compute the FOM approximant using (4.2).

5 Quadrature-Based Arnoldi Restarts for Matrix Functions with Randomized Sketching

Suppose we have a Arnoldi-like decomposition as in (4.1) where \mathbf{B}_m is a basis for the Krylov subspace $\mathcal{K}_m(\mathbf{A}; \mathbf{b})$ and $\mathbf{b} = \mathbf{b}_1\beta$. Instead of requiring \mathbf{b}_{m+1} to be orthogonal to \mathbf{B}_m , we require the sketched vector $\mathbf{S}\mathbf{b}_{m+1}$ to be orthogonal to the sketched Krylov subspace $\mathbf{S}\mathcal{K}_m(\mathbf{A}; \mathbf{b})$, i.e., $(\mathbf{S}\mathbf{B}_m)^*\mathbf{S}\mathbf{b}_{m+1} = \mathbf{0}$.

Consider now the sFOM approximant defined in (3.4). Denote $\mathbf{G}_m = (\mathbf{S}\mathbf{B}_m)^\dagger(\mathbf{S}\mathbf{A}\mathbf{B}_m)$. Using

$$\mathbf{S}\mathbf{A}\mathbf{B}_m = \mathbf{S}\mathbf{B}_m\mathbf{H}_m + \mathbf{S}\mathbf{b}_{m+1}h_{m+1,m}\mathbf{e}_m^\top$$

and $\mathbf{S}\mathbf{b}_{m+1} \perp \mathbf{S}\mathbf{B}_m$, we have $\mathbf{G}_m = \mathbf{H}_m$. Therefore, the sFOM approximant reduces to

$$\hat{\mathbf{f}}_m = \mathbf{B}_m f(\mathbf{H}_m)(\mathbf{S}\mathbf{B}_m)^\dagger(\mathbf{S}\mathbf{b}) = \mathbf{B}_m f(\mathbf{H}_m)\mathbf{e}_1\beta. \quad (5.1)$$

Suppose p_{m-1} is the unique polynomial of degree at most $m-1$ that interpolates f at the eigenvalues of \mathbf{G}_m , then $f(\mathbf{G}_m) = p_{m-1}(\mathbf{G}_m)$. Then using Lemma 4.1 again, we have

$$\hat{\mathbf{f}}_m = \mathbf{B}_m f(\mathbf{G}_m)\mathbf{e}_1\beta = \mathbf{B}_m p_{m-1}(\mathbf{G}_m)\mathbf{e}_1\beta = \mathbf{B}_m p_{m-1}(\mathbf{H}_m)\mathbf{e}_1\beta = p_{m-1}(\mathbf{A})\mathbf{b}_1\beta = p_{m-1}(\mathbf{A})\mathbf{b}.$$

Again, we can give an analogue result of Theorem 4.2 for the sFOM approximant.

Theorem 5.1 (Error of the sFOM approximation when $\mathbf{S}\mathbf{b}_{m+1}$ is orthogonal to $\mathbf{S}\mathbf{B}_m$). *Let (4.1) be a given Arnoldi-like decomposition where \mathbf{B}_m is a basis for the Krylov subspace $\mathcal{K}_m(\mathbf{A}; \mathbf{b})$ and $\mathbf{b} = \beta\mathbf{b}_1$. Suppose f has an integral representation as in (2.3). Denote $\hat{\mathbf{f}}_m = \mathbf{B}_m f(\mathbf{G}_m)\mathbf{e}_1\beta$ the sFOM approximation to $f(\mathbf{A})\mathbf{b}$ where $\mathbf{G}_m = (\mathbf{S}\mathbf{B}_m)^\dagger(\mathbf{S}\mathbf{A}\mathbf{B}_m)$. Let $\text{spec}(\mathbf{G}_m) = \{\theta_1, \dots, \theta_m\} \subset \Omega$, $\phi_m(t) = (t - \theta_1) \cdots (t - \theta_m)$ and $\gamma_m = \prod_{j=1}^m h_{j+1,j}$. If $\mathbf{S}\mathbf{b}_{m+1} \perp \mathbf{S}\mathbf{B}_m$, then the error of Arnoldi approximation can be expressed as*

$$f(\mathbf{A})\mathbf{b} - \hat{\mathbf{f}}_m = \gamma_m \beta \int_{\Gamma} \frac{g(t)}{\phi_m(t)} (t\mathbf{I} - \mathbf{A})^{-1} dt \mathbf{b}_{m+1} =: \text{err}_m(\mathbf{A})\mathbf{b}_{m+1}.$$

provided the integral exists. Therefore, the error function is given by

$$\text{err}_m(z) = \gamma_m \beta \int_{\Gamma} \frac{g(t)}{\phi_m(t)} (t - z)^{-1} dt.$$

Proof. Using Lemma 2.3, we have

$$p_{m-1}(z) = \int_{\Gamma} \left(1 - \frac{\phi_m(z)}{\phi_m(t)}\right) \frac{g(t)}{t - z} dt,$$

where $\phi_m(z) = (z - \theta_1) \cdots (z - \theta_m)$. Therefore,

$$f(z) - p_{m-1}(z) = \int_{\Gamma} \frac{\phi_m(z)}{\phi_m(t)} \frac{g(t)}{t - z} dt$$

and

$$f(\mathbf{A})\mathbf{b} - \hat{\mathbf{f}}_m = \int_{\Gamma} \frac{\phi_m(\mathbf{A})}{\phi_m(t)} (t\mathbf{I} - \mathbf{A})^{-1} g(t) dt \mathbf{b}.$$

Note that by Lemma 4.1, we have $\phi_m(\mathbf{A})\mathbf{b} = \phi_m(\mathbf{A})\mathbf{b}_1\beta = \mathbf{B}_m \phi_m(\mathbf{H}_m)\mathbf{e}_1\beta + \mathbf{b}_{m+1}\gamma_m\beta = \mathbf{b}_{m+1}\gamma_m\beta$. Then,

$$f(\mathbf{A})\mathbf{b} - \hat{\mathbf{f}}_m = \gamma_m \beta \int_{\Gamma} \frac{g(t)}{\phi_m(t)} (t\mathbf{I} - \mathbf{A})^{-1} dt \mathbf{b}_{m+1}.$$

This completes the proof. \square

5.1 Truncated Arnoldi Case

If \mathbf{B}_m is generated by a truncated-Arnoldi process, we do not necessarily have $\mathbf{S}\mathbf{b}_{m+1} \perp \mathbf{S}\mathbf{B}_m$. Therefore, we conduct an additional orthogonalization step. Suppose $\mathbf{c} \in \mathbb{C}^m$ such that $\mathbf{b}_{m+1} = \hat{\mathbf{b}}_{m+1}\alpha + \mathbf{B}_m\mathbf{c}$, such that $\hat{\mathbf{b}}_{m+1} \perp \mathbf{S}\mathbf{B}_m$. Such \mathbf{c} can be obtained by $\mathbf{c} = (\mathbf{S}\mathbf{B}_m)^\dagger(\mathbf{S}\mathbf{b}_{m+1})$. Then the Arnoldi-like decomposition (4.1) can be modified as

$$\begin{aligned}\mathbf{AB}_m &= \mathbf{B}_m\mathbf{H}_m + \mathbf{b}_{m+1}h_{m+1,m}\mathbf{e}_m^\top \\ &= \mathbf{B}_m\mathbf{H}_m + (\hat{\mathbf{b}}_{m+1} + \mathbf{B}_m\mathbf{c})h_{m+1,m}\mathbf{e}_m^\top \\ &= \mathbf{B}_m(\mathbf{H}_m + \mathbf{c}h_{m+1,m}\mathbf{e}_m^\top) + \hat{\mathbf{b}}_{m+1}h_{m+1,m}\mathbf{e}_m^\top \\ &= \mathbf{B}_m\hat{\mathbf{H}}_m + \hat{\mathbf{b}}_{m+1}h_{m+1,m}\mathbf{e}_m^\top.\end{aligned}\tag{5.2}$$

Then, the sFOM approximation is

$$\hat{\mathbf{f}}_m = \mathbf{B}_m f(\hat{\mathbf{H}}_m)(\mathbf{S}\mathbf{B}_m)^\dagger(\mathbf{S}\mathbf{b}) = \mathbf{B}_m f(\hat{\mathbf{H}}_m)\mathbf{e}_1\beta.\tag{5.3}$$

5.2 Sketched Arnoldi Case

If \mathbf{B}_m is generated by a truncated-Arnoldi process, we already have $\mathbf{S}\mathbf{b}_{m+1} \perp \mathbf{S}\mathbf{B}_m$. Thus we can use the matrix \mathbf{H}_m in the Arnoldi-like decomposition 4.1 directly and

$$\hat{\mathbf{f}}_m = \mathbf{B}_m f(\mathbf{H}_m)\mathbf{e}_1\beta.\tag{5.4}$$

The corresponding functions about the above discussions are:

- “funm_quad_sfom_last_orth_tarnoldi”
- “funm_quad_sfom_last_orth_sarnoldi”

6 Monitoring Basis Using Random Sketching

The idea of this method is, instead of fixing the dimension m of the Krylov space, we determine m adaptively. More specifically, we generate a basis \mathbf{B}_m adaptively by the truncated Arnoldi process. An sketching matrix \mathbf{S} is chosen to monitor the condition number of the sketched basis $\mathbf{S}\mathbf{B}_m$ during the process. Once the ill-conditionness is detected, the Arnoldi process is halted and we do an update of the last vector \mathbf{b}_{m+1} to ensure it is orthogonal to \mathbf{B}_m , as stated in Section 4. Then, we use the FOM approximation (4.2) and restart under the guidance of Theorem 4.2. See Algorithm 6 for the Arnoldi process.

The corresponding functions about the above discussions are:

- “tarnoldi_last_orth_adaptive”
- “funm_quad_ada_fom_last_orth_tarnoldi”

Algorithm 6 Adaptive truncated Arnoldi process using a random sketching monitor.

Input: Matrix \mathbf{A} , vector \mathbf{b} , truncation parameter t , tolerance of condition number τ , max dimension of the Krylov subspace m_{\max} .

Output: Subspace dimension m , basis $\mathbf{B}_m \in \mathbb{C}^{N \times m}$ for the Krylov subspace $\mathcal{K}_m(\mathbf{A}; \mathbf{b})$.

```

1: Set  $s = s_0 = 30$ .
2: Draw a sketching matrix  $\mathbf{S} \in \mathbb{C}^{s \times N}$ .
3:  $\mathbf{b}_1 = \mathbf{b}/\|\mathbf{b}\|$ .
4:  $\mathbf{B}_1 = [\mathbf{b}_1]$ ,  $\mathbf{P}_1 = [\mathbf{S}\mathbf{b}_1]$ .
5: for  $j = 1, \dots, m_{\max}$  do
6:    $\mathbf{w}_j = \mathbf{A}\mathbf{b}_j$ .
7:   for  $i = \max(1, j - t + 1), \dots, j$  do
8:      $h_{i,j} = \mathbf{b}_i^* \mathbf{w}_j$ .
9:      $\mathbf{w}_j = \mathbf{w}_j - \mathbf{b}_i h_{i,j}$ .
10:    end for
11:     $h_{j+1,j} = \|\mathbf{w}_j\|$ .
12:    if  $h_{j+1,j} = 0$  then
13:      Stop.
14:    end if
15:     $\mathbf{b}_{j+1} = \mathbf{w}_j/h_{j+1,j}$ .
16:     $\mathbf{B}_{j+1} = [\mathbf{B}_j, \mathbf{b}_{j+1}]$ ,  $\mathbf{P}_{j+1} = [\mathbf{P}_j, \mathbf{S}\mathbf{b}_{j+1}]$ .
17:    if  $\text{cond}(\mathbf{P}_{j+1}) > \tau$  then
18:      Break
19:    end if
20:    if  $s < 2j$  then
21:      Draw a sketching matrix  $\mathbf{S}_{\text{incr}} \in \mathbb{C}^{s_0 \times N}$ .
22:      Update  $\mathbf{S} \leftarrow \begin{bmatrix} \mathbf{S} \\ \mathbf{S}_{\text{incr}} \end{bmatrix}$  and  $\mathbf{P}_{j+1} \leftarrow \begin{bmatrix} \mathbf{P}_{j+1} \\ \mathbf{S}_{\text{incr}} \mathbf{B}_{j+1} \end{bmatrix}$ 
23:       $s \leftarrow s + s_0$ .
24:    end if
25:  end for
26:   $m = j$ .
27:  Update  $\mathbf{b}_{m+1}$  to be orthogonal to  $\mathbf{B}_m$ .
28:   $\mathbf{B}_m = [\mathbf{b}_1, \dots, \mathbf{v}_m]$ .

```

7 Numerical Experiments

Through all experiments, we set the sketching dimension to be roughly $2m$ where m is the dimension of the Krylov subspace. For the truncated Arnoldi process, we set the truncation parameter to be 5. We use the Gaussian random matrix as the sketching matrix \mathbf{S} . The tolerance for quadrature rule is set to be 10^{-7} and the stopping accuracy for the restart loop is set to be 10^{-8} . The maximum number of restart is set to be 15 and the maximum number of quadrature point updates is set to be 1024. We use Algorithm 2 as a benchmark, comparing both runtime and accuracy.

Here are some notations of the methods:

- FOM-t: using the truncated Arnoldi process to generate the basis \mathbf{B}_m and compute the FOM approximant.
- FOM-s: using the sketched Arnoldi process to generate the basis \mathbf{B}_m and compute the FOM approximant.
- sFOM-t: using the truncated Arnoldi process to generate the basis \mathbf{B}_m and compute the sketched FOM approximant.
- sFOM-s: using the sketched Arnoldi process to generate the basis \mathbf{B}_m and compute the sketched FOM approximant.
- aFOM-t: using the adaptive truncated Arnoldi process to generate the basis \mathbf{B}_m and compute the FOM approximant.

7.1 Matrix Exponential

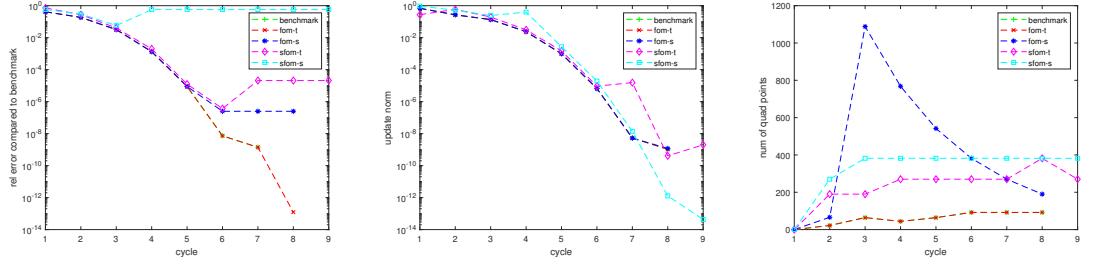
The second example is from Section 5.3 of [2], where the matrix function is given by $f(\mathbf{A}) = \exp(s\mathbf{A})$ and the matrices $\mathbf{A} \in \mathbb{R}^{N \times N}$ correspond to the standard finite difference discretization of a two-dimensional convection-diffusion operator on the unit square with constant convection field and different convection parameters ν . The case $\nu = 0$ corresponds to a symmetric problem and for increasing ν the non-normal matrix \mathbf{A} has eigenvalues with large imaginary parts. We choose $s = 2 \cdot 10^{-3}$ and use 500 discretization points in both spatial directions, resulting in a matrix of size $N = 500^2$ and ν is set to be 100.

We first set $m = 70$. The relavent results are shown in Table 7.1 and Figure 7.1.

Table 7.1: Results for matrix exponential, $m = 70$.

Method	total iter	rel error (to benchmark)	time
benchmark	8	0.0000e+00	1.2837e+01
FOM-t	8	1.2196e-13	5.3486e+00
sFOM-t	9	2.0862e-05	7.5628e+00
FOM-s	8	2.4953e-07	2.0459e+01
sFOM-s	9	5.7011e-01	2.1859e+01

Then we set $m = 200$ and $m_{\max} = 500$. The relavent results are shown in Table 7.2 and Figure 7.2. Subspace dimensions for each restart are plotted in Figure 7.3.



(a) Relative error to benchmark. (b) Norm of update. (c) Number of quadrature points.

Figure 7.1: Results for matrix exponential, $m = 70$.

Table 7.2: Results for matrix exponential, $m = 200$, $m_{\max} = 500$.

Method	total iter	rel error (to benchmark)	time
benchmark	5	0.0000e+00	8.7917e+01
FOM-t	5	7.8806e-11	2.3158e+01
sFOM-t	5	1.6970e-10	2.0230e+01
FOM-s	5	1.2491e-10	1.3165e+02
sFOM-s	5	3.0222e-03	1.2690e+02
aFOM-t	3	1.2496e-10	2.1381e+02

7.2 Matrix Inverse Square Root

The matrix function is given by $f(\mathbf{A}) = \mathbf{A}^{-1/2}$ and the matrices $\mathbf{A} \in \mathbb{R}^{N \times N}$ correspond to the 2d Laplacian plus some non-Hermitian matrix. We use 500 discretization points in both spatial directions, resulting in a matrix of size $N = 500^2$.

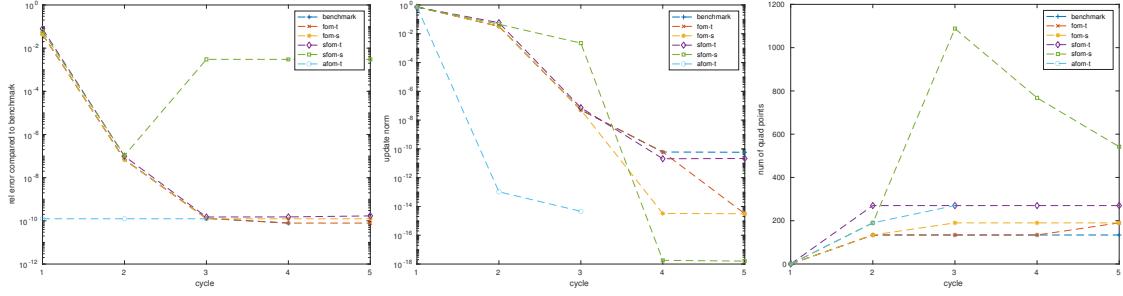
The relavent results are shown in Table 7.3 and Figure 7.4 for $m = 70$ and Table 7.4 and Figure 7.5 for $m = 200$. Subspace dimensions for each restart are plotted in Figure 7.6.

Table 7.3: Results for matrix inverse square root, $m = 70$.

Method	total iter	rel error (to benchmark)	time
benchmark	15	0.0000e+00	2.4069e+01
FOM-t	15	2.2179e-11	9.9631e+00
sFOM-t	15	4.2022e-01	1.4124e+01
FOM-s	15	1.3817e-08	3.7996e+01
sFOM-s	15	1.2756e-01	3.8290e+01

7.3 Matrix Log

The matrix function is given by $f(\mathbf{A}) = \mathbf{A}^{-1/2}$ and the matrices $\mathbf{A} \in \mathbb{R}^{N \times N}$ correspond to the 2d Laplacian plus some non-Hermitian matrix. We use 500 discretization points in both spatial



(a) Relative error to benchmark. (b) Norm of update. (c) Number of quadrature points.

Figure 7.2: Results for matrix exponential, $m = 200$.

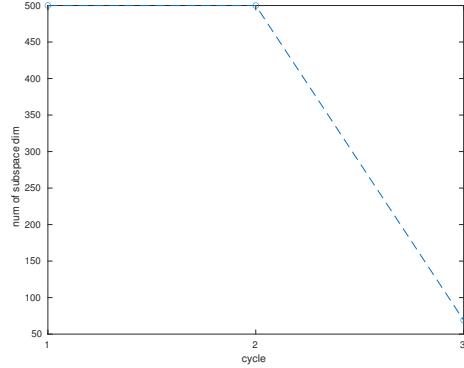


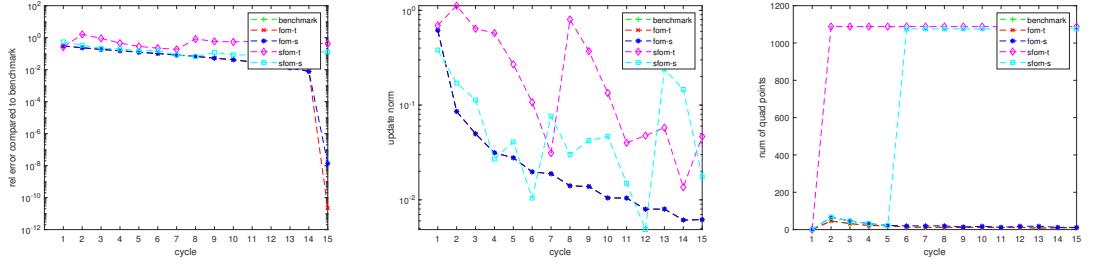
Figure 7.3: Subspace dimension in matrix exponential, $m = 200$, $m_{\max} = 500$.

Table 7.4: Results for matrix inverse square root, $m = 200$, $m_{\max} = 500$.

Method	total iter	rel error (to benchmark)	time
benchmark	15	0.0000e+000	2.1312e+02
FOM-t	15	4.4630e-11	5.9961e+01
sFOM-t	15	2.5044e-07	5.6820e+01
FOM-s	15	1.9224e-09	3.3751e+02
sFOM-s	15	9.2112e-07	3.3008e+02
aFOM-t	8	6.5897e-07	7.4505e+02

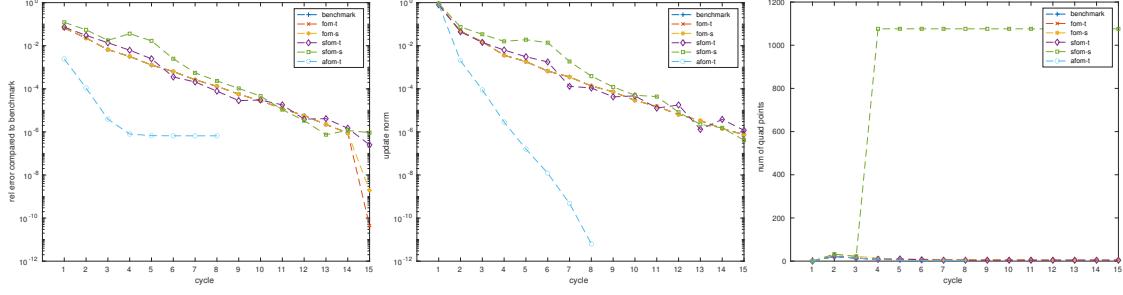
directions, resulting in a matrix of size $N = 500^2$.

The relavent results are shown in Table 7.5 and Figure 7.7 for $m = 70$ and Table 7.6 and Figure 7.8 for $m = 200$.



(a) Relative error to benchmark. (b) Norm of update. (c) Number of quadrature points.

Figure 7.4: Results for matrix inverse square root, $m = 70$.



(a) Relative error to benchmark. (b) Norm of update. (c) Number of quadrature points.

Figure 7.5: Results for matrix inverse square root, $m = 200$.

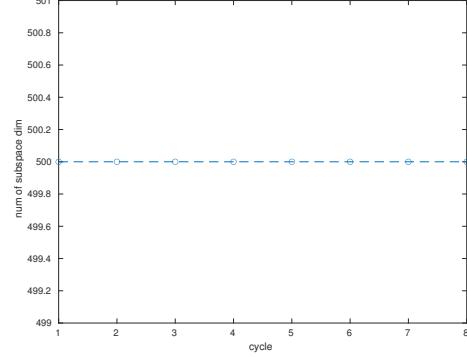


Figure 7.6: Subspace dimension in inverse square root, $m = 200$, $m_{\max} = 500$.

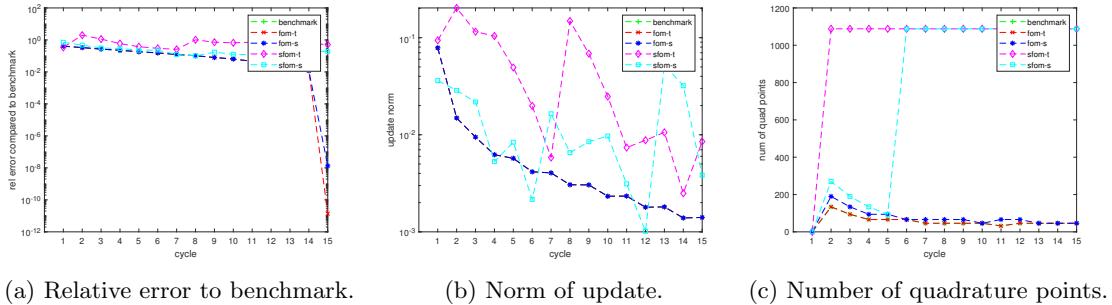
8 What is next?

References

- [1] A. CORTINOVIS, D. KRESSNER, AND Y. NAKATSUKASA, *Speeding up krylov subspace methods*

Table 7.5: Results for matrix logarithm, $m = 70$.

Method	total iter	rel error (to benchmark)	time
benchmark	15	0.0000e+000	2.4302e+01
FOM-t	15	1.3672e-11	1.0048e+01
sFOM-t	15	5.1258e-01	1.8253e+01
FOM-s	15	1.3425e-08	3.8098e+01
sFOM-s	15	1.9029e-01	4.1083e+01



(a) Relative error to benchmark. (b) Norm of update. (c) Number of quadrature points.

Figure 7.7: Results for matrix logarithm, $m = 70$.

Table 7.6: Results for matrix logarithm, $m = 200$, $m_{\max} = 500$.

Method	total iter	rel error (to benchmark)	time
FOM-t	15	1.2832e-08	6.7709e+01
sFOM-t	15	3.1544e-07	6.1022e+01
FOM-s	15	2.2682e-08	3.9118e+02
sFOM-s	15	9.9827e-07	3.8319e+02
aFOM-t	8	7.2540e-07	8.0678e+02

for computing $\mathbf{f}(\mathbf{A})\mathbf{b}$ via randomization, SIAM J. Matrix Anal. Appl., 45 (2024), pp. 619–633.

- [2] A. FROMMER, S. GÜTTEL, AND M. SCHWEITZER, Efficient and stable Arnoldi restarts for matrix functions based on quadrature, SIAM J. Matrix Anal. Appl., 35 (2014), pp. 661–683.
- [3] S. GÜTTEL AND M. SCHWEITZER, Randomized sketching for Krylov approximations of large-scale matrix functions, SIAM J. Matrix Anal. Appl., 44 (2023), pp. 1073–1095.

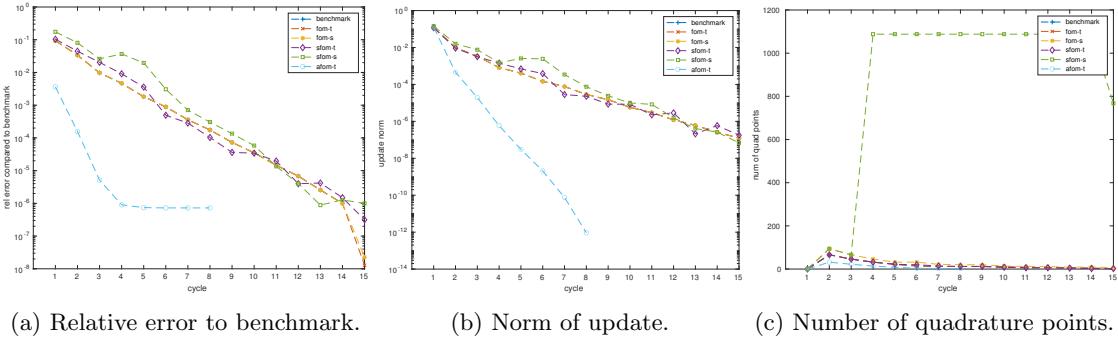


Figure 7.8: Results for matrix logarithm, $m = 200$.

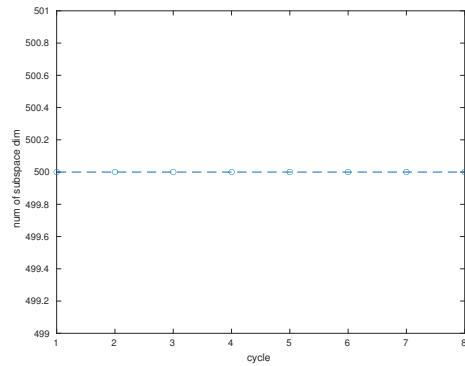


Figure 7.9: Subspace dimension in log, $m = 200$, $m_{\max} = 500$.