TECHNISCHE UNIVERSITÄT BERGAKADEMIE FREIBERG

Institut für Numerische Mathematik und Optimierung

Implementation of Restarted Krylov Method for the Approximation of Matrix Functions

Stefan Güttel

joint work with

Martin Afanasjew, Michael Eiermann & Oliver G. Ernst

Harrachov, August 2007

Problem

Given: $A \in \mathbb{C}^{n \times n}$, $\boldsymbol{b} \in \mathbb{C}^n$, $\|\boldsymbol{b}\|_2 = 1$, f nice.

Sought: $f(A)\boldsymbol{b}$.

Outline

- Arnoldi approximation
- Restarting
- Convergence
- Stopping criteria

Arnoldi Approximation

Arnoldi process applied to A with initial vector $\boldsymbol{b}$ yields
Arnoldi (Lanczos) decomposition

$$AV_m = V_m H_m + \eta_{m+1,m} \boldsymbol{v}_{m+1} \boldsymbol{e}_m^T,$$

where

$$V_m = [\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_m] \in \mathbb{C}^{n\times m}, \quad \{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_m\} \text{ ON basis of } \mathscr{K}_m(A, \boldsymbol{b}),$$
$$\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{m+1}\} \text{ ON basis of } \mathscr{K}_{m+1}(A, \boldsymbol{b}),$$
$$\boldsymbol{v}_1 = \boldsymbol{b},$$
$$H_m = [\eta_{i,j}] \in \mathbb{C}^{m\times m} \quad \text{upper Hessenberg matrix},$$
$$\boldsymbol{e}_m = [0, \ldots, 0, 1]^T \in \mathbb{R}^m.$$

Arnoldi approximation of $f(A)\boldsymbol{b}$:

$$f(A)\boldsymbol{b} \approx \boldsymbol{f}_m := V_m f(H_m)\boldsymbol{e}_1$$

[Druskin & Knizhnerman, 1989],
[Gallopoulos & Saad, 1992],
[Saad, 1992],
[Hochbruck & Lubich, 1995],
[Philippe & Sidje, 1995, 1998],
[Frommer & Simoncini, 2006].

Interpretation: Let $p_{m-1} \in \mathscr{P}_{m-1}$ be a (the) polynomial which interpolates f at the eigenvalues of H_m. Then

$$f(H_m) = p_{m-1}(H_m) \qquad \text{by definition of } f$$
$$\boldsymbol{f}_m = V_m p_{m-1}(H_m)\boldsymbol{e}_1 = p_{m-1}(A)\boldsymbol{b} \qquad \text{since } p_{m-1} \in \mathscr{P}_{m-1}.$$

The Arnoldi approximation can therefore be seen as an interpolation process where the nodes are the eigenvalues of H_m.

$$\boxed{\boldsymbol{f}_m = V_m f(H_m)\boldsymbol{e}_1}$$

Advantage: avoids A (only $\boldsymbol{v} \to A\boldsymbol{v}$), avoids explicit interpolation, requires only evaluation of $f(H_m)$ for (small) matrix H_m.

Drawback: high computational cost for Arnoldi decomposition (in nonsymm. case), requires basis V_m (extensive storage)

in Hermitian case: 2-pass Lanczos

- first pass of Lanczos to determine H_m,
- compute $f(H_m)\boldsymbol{e}_1$,
- second pass of Lanczos to determine the linear combination of $\boldsymbol{v}_j$ with coefficients from $f(H_m)\boldsymbol{e}_1$.

(doubles the number of matrix-vector products)

Restarting the Arnoldi Approximation

Consider two cycles of Arnoldi

$$AV_1 = V_1 H_1 + \eta_2 \boldsymbol{v}_{m+1} \boldsymbol{e}_m^T, \qquad V_1(:,1) = \boldsymbol{b}$$
$$AV_2 = V_2 H_2 + \eta_3 \boldsymbol{v}_{2m+1} \boldsymbol{e}_m^T, \qquad V_2(:,1) = \boldsymbol{v}_{m+1}.$$

Since columns of $\widehat{V}_2 := [V_1, V_2]$ form (nonorthogonal) basis of $\mathscr{K}_{2m}(A, \boldsymbol{b})$, we can combine to Arnoldi-like decomposition

$$A\widehat{V}_2 = \widehat{V}_2 \widehat{H}_2 + \eta_3 \boldsymbol{v}_{2m+1} \boldsymbol{e}_{2m}^T,$$

$$\widehat{H}_2 := \begin{bmatrix} H_1 & O \\ E_2 & H_2 \end{bmatrix}, \qquad E_2 := \eta_2 \boldsymbol{e}_1 \boldsymbol{e}_m^T \in \mathbb{R}^{m \times m}.$$

Arnoldi-like approximation: $\widehat{\boldsymbol{f}}_2 := \widehat{V}_2 f(\widehat{H}_2) \boldsymbol{e}_1$.

Interpretation: $\widehat{\boldsymbol{f}}_2 = p_{2m-1}(A)\boldsymbol{b}$, where p_{2m-1} interpolates f at $\Lambda(\widehat{H}_2) = \Lambda(H_1) \cup \Lambda(H_2)$.

After k cycles: $A\widehat{V}_k = \widehat{V}_k \widehat{H}_k + \eta_{k+1}\boldsymbol{v}_{km+1}\boldsymbol{e}_{km}^T$, where $\widehat{V}_k := [V_1, V_2, \ldots, V_k]$,

$$\widehat{H}_k := \begin{bmatrix} H_1 & & & \\ E_2 & H_2 & & \\ & \ddots & \ddots & \\ & & E_k & H_k \end{bmatrix}, \qquad E_j := \eta_j \boldsymbol{e}_1 \boldsymbol{e}_m^T \in \mathbb{R}^{m\times m}.$$

With

$$\widehat{F}_k := f(\widehat{H}_k) = \begin{bmatrix} F_{1,1} & & & \\ F_{2,1} & F_{2,2} & & \\ \vdots & \vdots & \ddots & \\ F_{k,1} & F_{k,2} & \ldots & F_{k,k} \end{bmatrix}, \quad \text{where } F_{j,j} = f(H_j),$$

we have

$$\widehat{\boldsymbol{f}}_k := \widehat{V}_k f(\widehat{H}_k)\boldsymbol{e}_1 = \underbrace{V_1 F_{1,1}\boldsymbol{e}_1}_{\text{first cycle}} + \underbrace{V_2 F_{2,1}\boldsymbol{e}_1}_{\text{second cycle}} + \cdots + \underbrace{V_k F_{k,1}\boldsymbol{e}_1}_{k\text{-th cycle}}.$$

Algorithm 1 [Eiermann & Ernst, 2006]

Given A, $\boldsymbol{b}$, $\|\boldsymbol{b}\| = 1$, f
$\boldsymbol{v}_1 := \boldsymbol{b}$, $\widehat{\boldsymbol{f}}_0 := \boldsymbol{0}$
For $k = 1, 2, \ldots$
  Compute Arnoldi decomposition $AV_k = V_k H_k + \eta_{k+1} \boldsymbol{v}_{km+1} \boldsymbol{e}_m^T$
    with initial vector $\boldsymbol{v}_{(k-1)m+1}$
  If $k = 1$
    $\widehat{H}_k := H_1$
  Else $\widehat{H}_k := \begin{bmatrix} \widehat{H}_{k-1} & O \\ \eta_k \boldsymbol{e}_1 \boldsymbol{e}_{(k-1)m}^T & H_k \end{bmatrix}$
  Update the approximation $\widehat{\boldsymbol{f}}_k := \widehat{\boldsymbol{f}}_{k-1} + V_k [f(\widehat{H}_k) \boldsymbol{e}_1]_{(k-1)m+1:km}$.

Note:
- Only V_k is needed in cycle k.
- $f(\widehat{H}_k)$ is computed –
  but only the last m entries of its first column are needed!

First idea: Compare blocks in the identity $\widehat{F}_k \widehat{H}_k = \widehat{H}_k \widehat{F}_k$ [Parlett],

$$\begin{bmatrix} F_{1,1} & & & \\ F_{2,1} & F_{2,2} & & \\ \vdots & \vdots & \ddots & \\ F_{k,1} & F_{k,2} & \cdots & F_{k,k} \end{bmatrix} \begin{bmatrix} H_1 & & & \\ E_2 & H_2 & & \\ & \ddots & \ddots & \\ & & E_k & H_k \end{bmatrix} = \begin{bmatrix} H_1 & & & \\ E_2 & H_2 & & \\ & \ddots & \ddots & \\ & & E_k & H_k \end{bmatrix} \begin{bmatrix} F_{1,1} & & & \\ F_{2,1} & F_{2,2} & & \\ \vdots & \vdots & \ddots & \\ F_{k,1} & F_{k,2} & \cdots & F_{k,k} \end{bmatrix}$$

we have for $j > \ell$,

$$F_{j,\ell} H_\ell - H_j F_{j,\ell} = E_j F_{j-1,\ell} - F_{j,\ell+1} E_{\ell+1}.$$

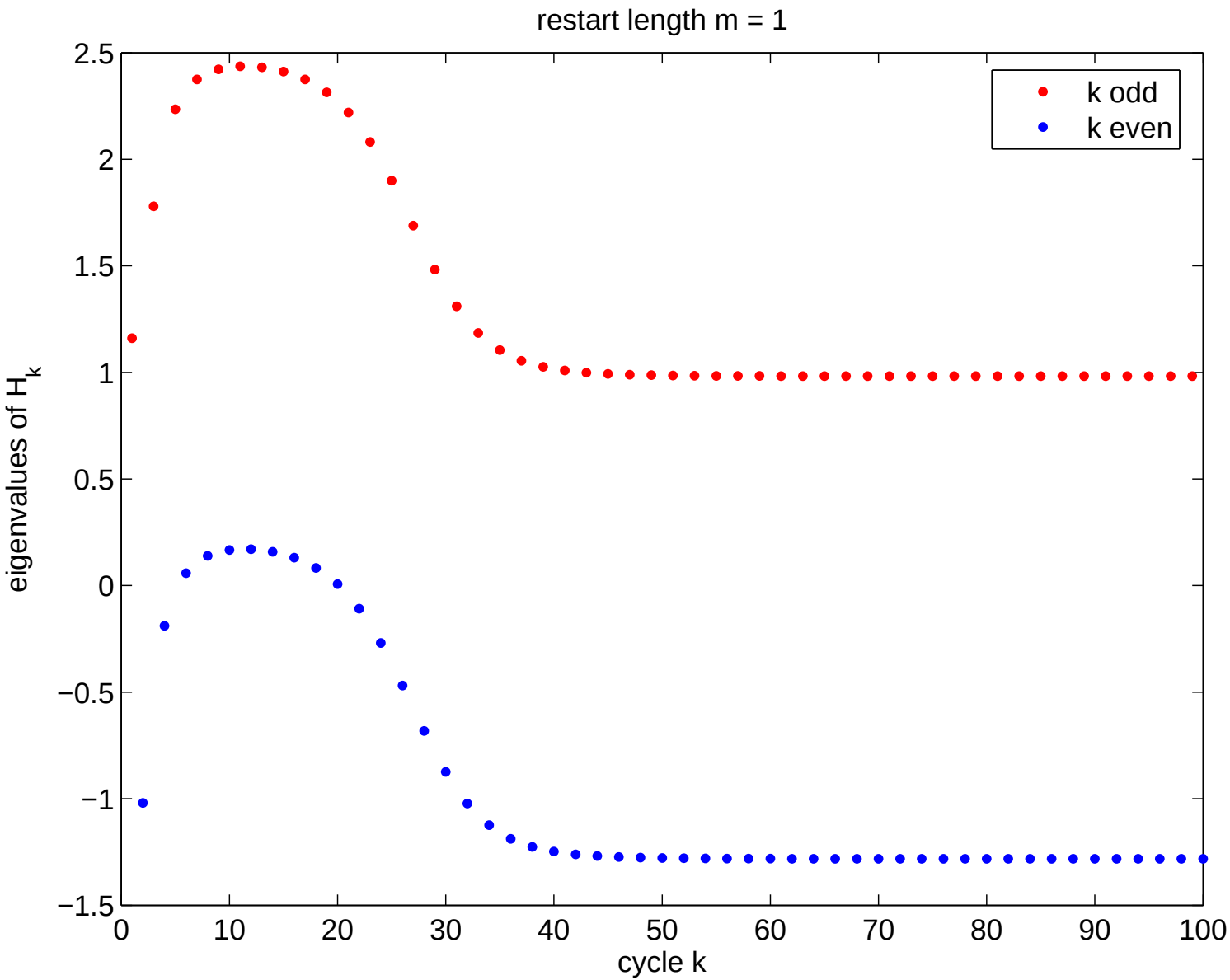
Since $F_{k,k} = f(H_k)$, can compute the last block row of $\widehat{F}_k$ recursively

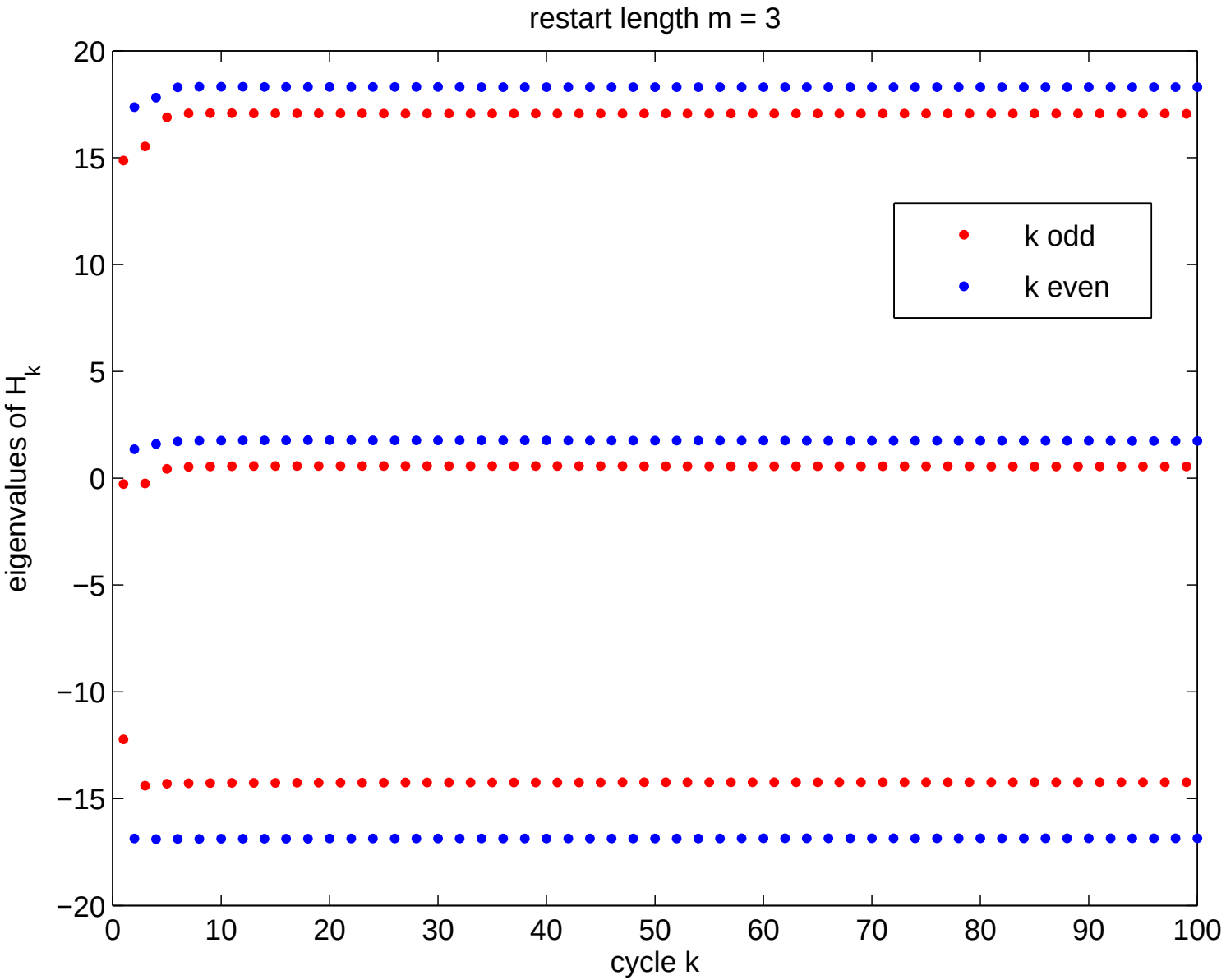
$$X H_{k-j} - H_k X = E_k F_{k-1,k-j} - F_{k,k-j+1} E_{k-j+1}, \quad j = 1, 2, \ldots, k-1,$$

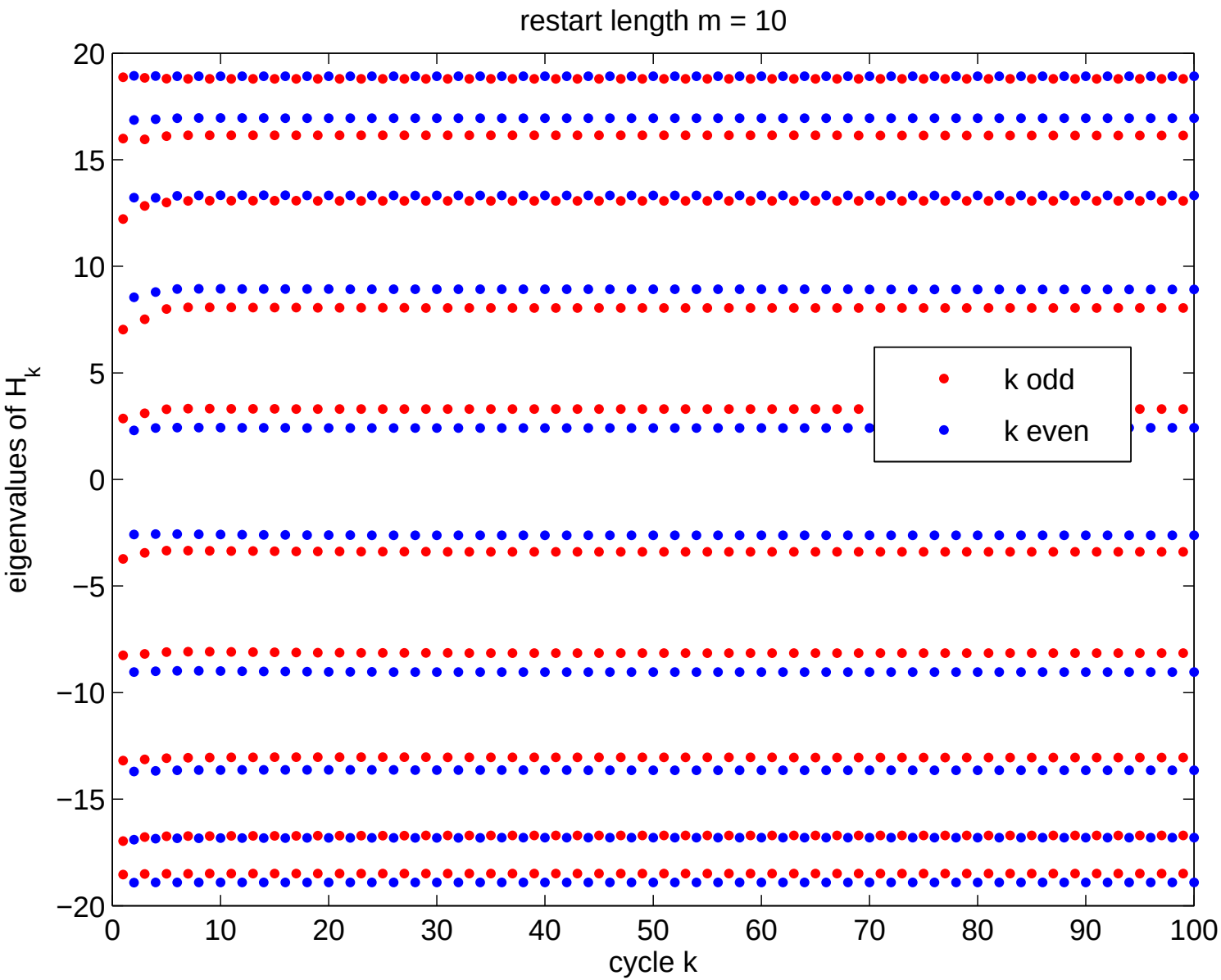
for $X = F_{k,k-j}$.

These Sylvester equations are

- easy to solve since the coefficients H_{k-j} and H_k are upper Hessenberg,
- ill-conditioned since the spectra of H_{k-j} and H_k are by no means well-separated...


restart length m = 1
k odd
k even
eigenvalues of H_k
cycle k
2.5
2
1.5
1
0.5
0
−0.5
−1
−1.5
0
10
20
30
40
50
60
70
80
90
100


restart length m = 3
eigenvalues of H_k
cycle k
k odd
k even
20
15
10
5
0
−5
−10
−15
−20
0
10
20
30
40
50
60
70
80
90
100


restart length m = 10
eigenvalues of H_k
cycle k
k odd
k even
20
15
10
5
0
−5
−10
−15
−20
0
10
20
30
40
50
60
70
80
90
100

Approximating f by rational function r in partial fraction form used by Gallopoulos & Saad [1992], Lopez & Simoncini [2006] for unrestarted case

$$f(\widehat{H}_k)\boldsymbol{e}_1 \approx r(\widehat{H}_k)\boldsymbol{e}_1 = \left[\alpha_0 I + \sum_{\ell=1}^{N} \alpha_\ell (\omega_\ell I - \widehat{H}_k)^{-1}\right] \boldsymbol{e}_1.$$

$$r(\widehat{H}_k)\boldsymbol{e}_1 = \alpha_0\, \boldsymbol{e}_1 + \sum_{\ell=1}^{N} \widehat{\boldsymbol{r}}_\ell, \quad \text{where}$$

$$(\omega_\ell I - \widehat{H}_k)\widehat{\boldsymbol{r}}_\ell = \begin{bmatrix} \omega_\ell I - H_1 & & & \\ -E_2 & \omega_\ell I - H_2 & & \\ & \ddots & \ddots & \\ & & -E_k & \omega_\ell I - H_k \end{bmatrix} \begin{bmatrix} \boldsymbol{r}_{\ell,1} \\ \boldsymbol{r}_{\ell,2} \\ \vdots \\ \boldsymbol{r}_{\ell,k} \end{bmatrix} = \begin{bmatrix} \boldsymbol{e}_1 \\ \boldsymbol{0} \\ \vdots \\ \boldsymbol{0} \end{bmatrix}$$

or, recursively,

$$\begin{aligned} (\omega_\ell I - H_1)\boldsymbol{r}_{\ell,1} &= \boldsymbol{e}_1, \\ (\omega_\ell I - H_j)\boldsymbol{r}_{\ell,j} &= E_j \boldsymbol{r}_{\ell,j-1}, \quad j = 2, \ldots, k. \end{aligned}$$

Algorithm 2 [Afanasjew, Eiermann, Ernst & G., 2007]

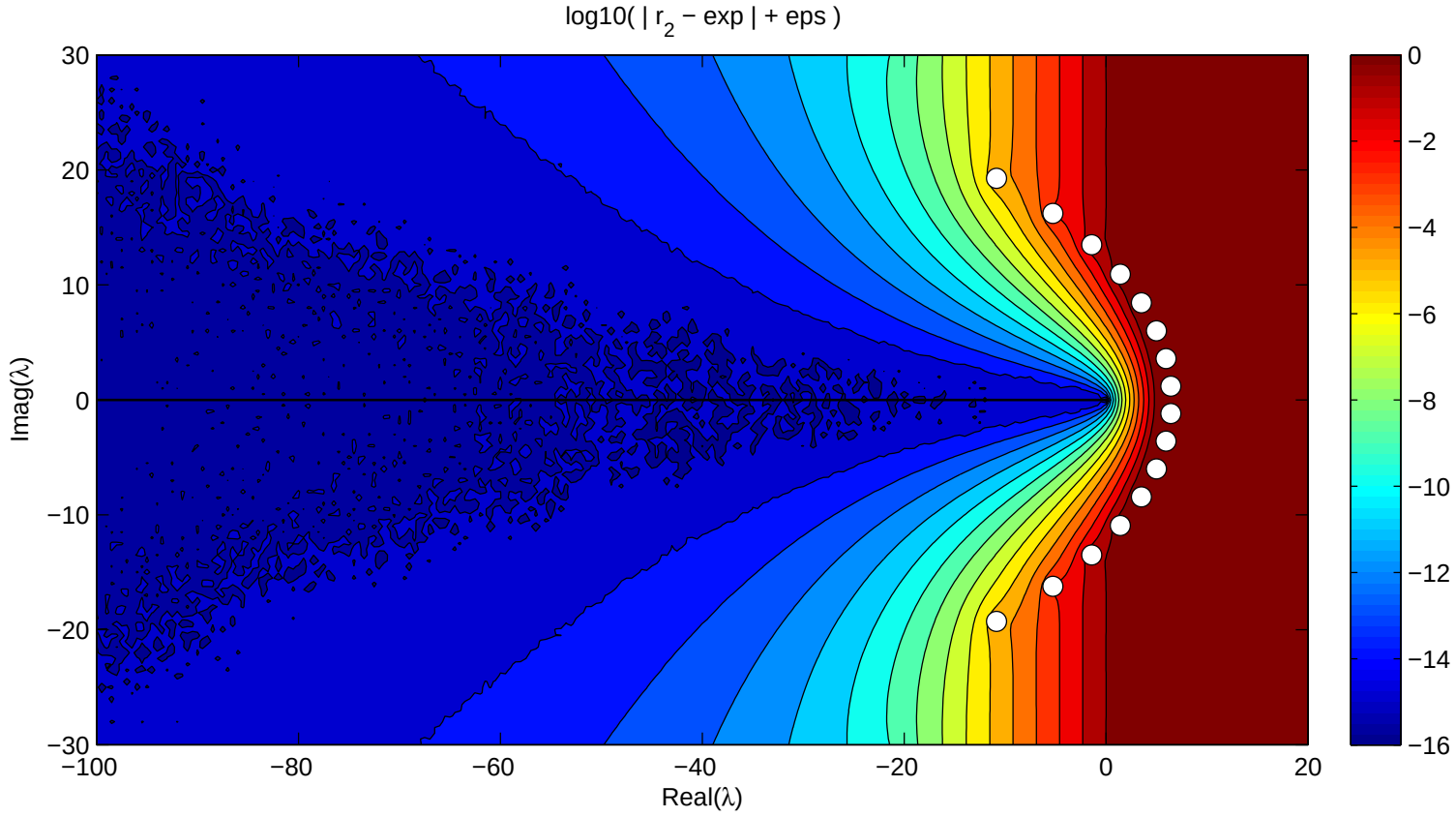
Given A, $\boldsymbol{b}$, $\|\boldsymbol{b}\| = 1$, $(\alpha_\ell, \omega_\ell)_{\ell=1}^N$
$\boldsymbol{v}_1 := \boldsymbol{b}$, $\widehat{\boldsymbol{f}}_0 := \alpha_0 \boldsymbol{b}$
For $k = 1, 2, \ldots$
    Compute Arnoldi decomposition $AV_k = V_k H_k + \eta_{k+1} \boldsymbol{v}_{km+1} \boldsymbol{e}_m^T$
        with initial vector $\boldsymbol{v}_{(k-1)m+1}$
    If $k = 1$
        For $\ell = 1, 2, \ldots, N$
            Solve $(\omega_\ell I - H_k)\boldsymbol{r}_{\ell,1} = \boldsymbol{e}_1$
    Else
        For $\ell = 1, 2, \ldots, N$
            Solve $(\omega_\ell I - H_k)\boldsymbol{r}_{\ell,k} = E_k \boldsymbol{r}_{\ell,k-1}$
    $\boldsymbol{h}_k := \sum_{\ell=1}^N \alpha_\ell \boldsymbol{r}_{\ell,k}$
    $\widehat{\boldsymbol{f}}_k := \widehat{\boldsymbol{f}}_{k-1} + V_k \boldsymbol{h}_k$.

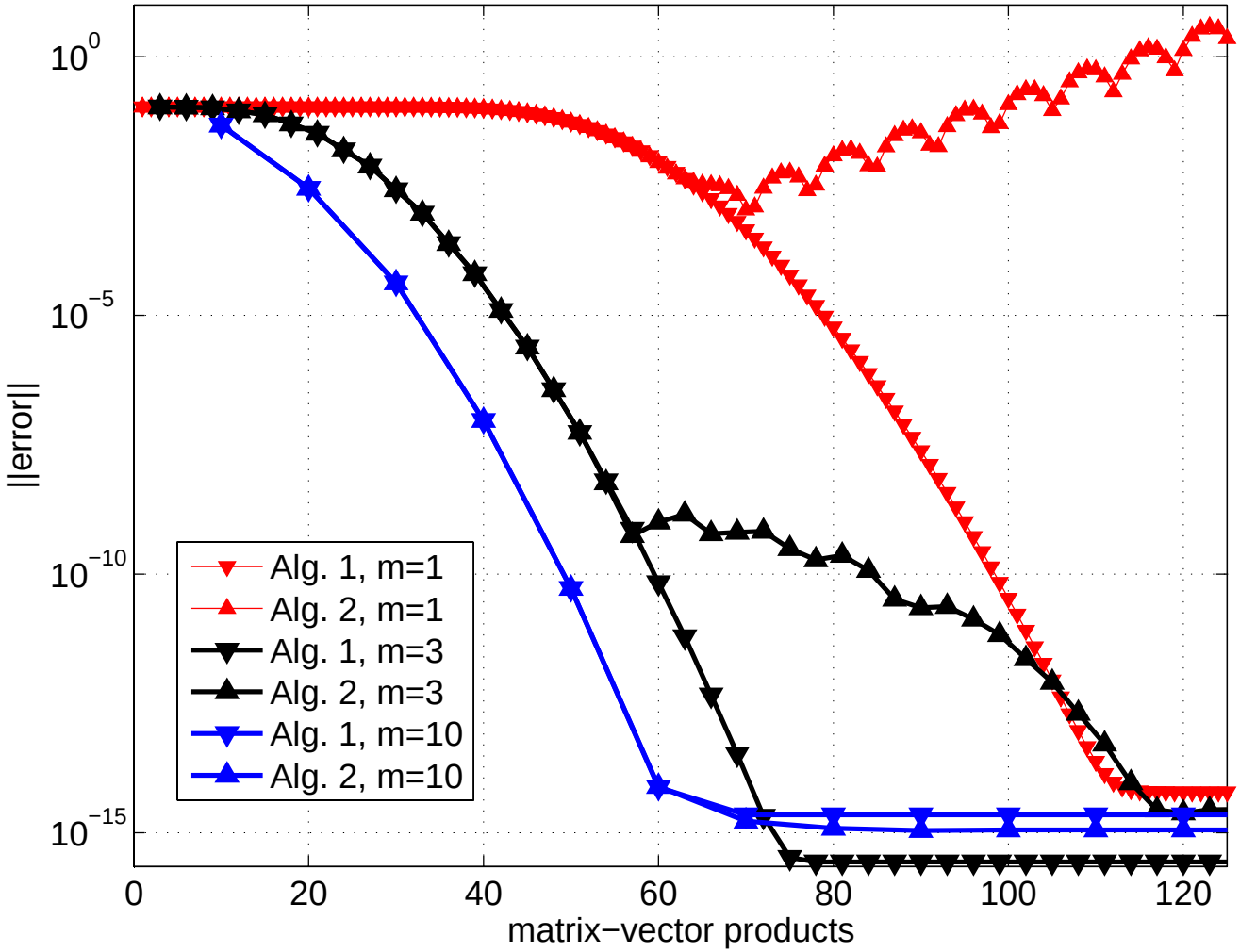
Note: Constant work and constant storage space per restart cycle.

Convergence Analysis

Model problem: $f = \exp,\ r_2 =$ CMV approximation,

$$A = \mathrm{diag}(-100, -99, \ldots, 0) \in \mathbb{R}^{101\times 101},\ \boldsymbol{b} = [1, 1, \ldots, 1]^T/\sqrt{101} \in \mathbb{R}^{101}.$$





Observation: Convergence history of Algorithm 2 shows two phases.

Algorithm 1: Approximates $\exp(\widehat{H}_k)$ by $r_1(\widehat{H}_k) = r(\widehat{H}_k/2^s)^{2^s}$, where r is a Padé-fraction (`expm`, [Higham, 2005]), r_1 depends on the argument and is not a rational function: if the argument is a scalar λ then $r_1(\lambda)$ is an accurate approximation to $\exp(\lambda)$ regardless of where in the complex plane λ is located; in particular, r_1 has no finite poles.

Algorithm 2: Approximates $\exp(\widehat{H}_k)$ by $r_2(\widehat{H}_k)$, where r_2 is a fixed rational function, namely the CMV approximation of type [16/16]. r_2 approximates $\exp$ well only on $(-\infty, 0]$ and has finite poles.

Main distinction: The restarted Arnoldi method converges superlinearly to an entire function (Algorithm 1) and converges/diverges linearly since r_2 has finite singularities (Algorithm 2).

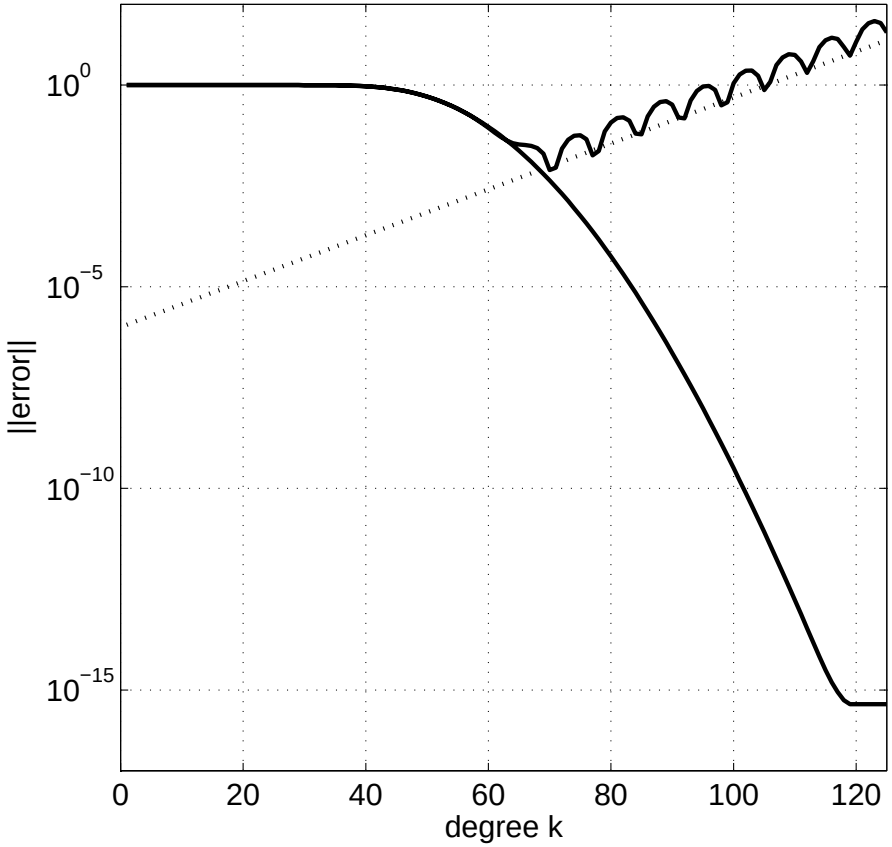
Restart length $m = 1$:

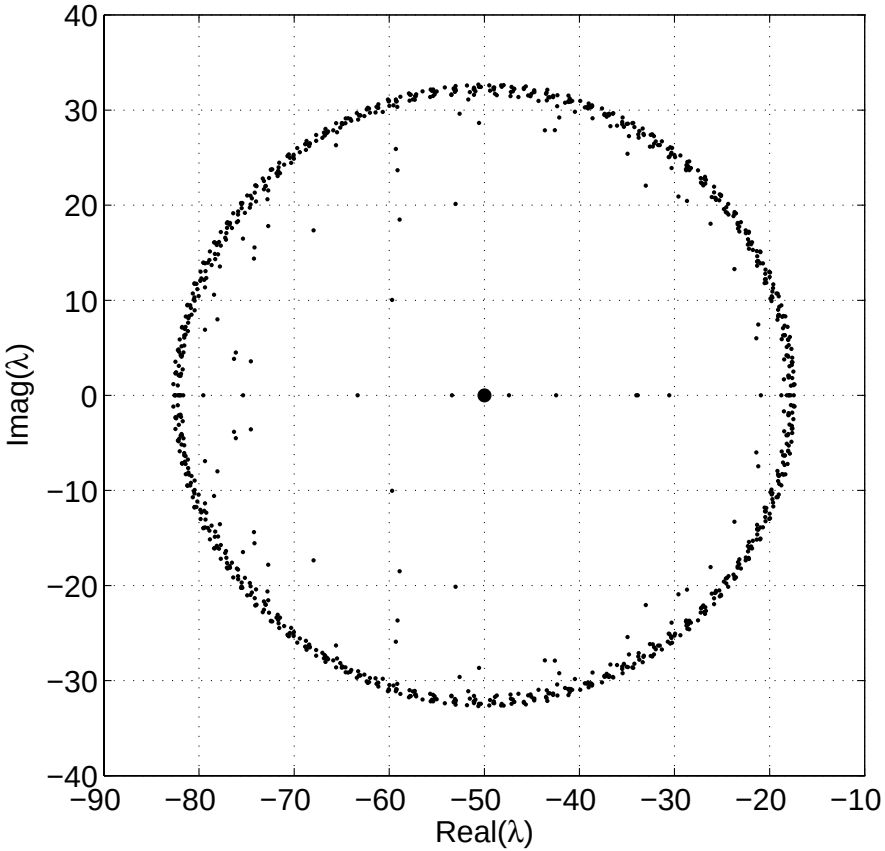
- $\widehat{\boldsymbol{f}}_k = p_{k-1}(A)\boldsymbol{b}$, where p_{k-1} interpolates $f = \exp$ or $f = r_2$ at the eigenvalues of $\widehat{H}_k$,
- $\widehat{H}_k$ is bidiagonal, all diagonal entries are equal to $\vartheta = -50$,
- the interpolating polynomials are therefore truncated Taylor series,
- the Taylor series for $\exp$ (Algorithm 1) converges everywhere, whereas the Taylor series for r_2 (Algorithm 2) converges only for

$$|\lambda + 50| < \min_{\omega = \text{pole of } r_2} |\omega + 50| \approx 44 < 50 = \max_{\lambda \in \Lambda(A)} |\lambda + 50|,$$

- Algorithm 2 therefore ultimately diverges like

$$\left[\max_{\lambda \in \Lambda(A)} |\lambda + 50|/44 \right]^k = (50/44)^k \approx 1.14^k.$$


10^{0}
10^{-5}
10^{-10}
10^{-15}
||error||
0
20
40
60
80
100
120
degree k


40
30
20
10
0
−10
−20
−30
−40
Imag(λ)
−90
−80
−70
−60
−50
−40
−30
−20
−10
Real(λ)

Restart length $m \geq 1$:

- $\widehat{H}_k$ is a block-bidiagonal matrix, where the diagonal blocks are symmetric tridiagonal,
- we observe

$$\lim_{j\to\infty} H_{2j-1} = \widetilde{H}_1 \quad \text{and} \quad \lim_{j\to\infty} H_{2j} = \widetilde{H}_2,$$

- the nodal sequence (the eigenvalues of $\widehat{H}_k$) $\vartheta_1, \vartheta_2, \vartheta_3, \ldots$ therefore has the property

$$\lim_{j\to\infty} \vartheta_{2mj+\nu} = \widetilde{\vartheta}_\nu \qquad \text{for } \nu = 1, 2, \ldots, 2m,$$

- by theorems of Walsh [1969] the convergence of the interpolation process is asymptotically determined by the lemniscates

$$L_\tau := \{\lambda \in \mathbb{C} : |w_{2m}(\lambda)| = \tau^{2m}\}, \quad \tau > 0, \quad w_{2m}(\lambda) = \textstyle\prod_{\nu=1}^{2m}(\lambda - \widetilde{\vartheta}_\nu),$$

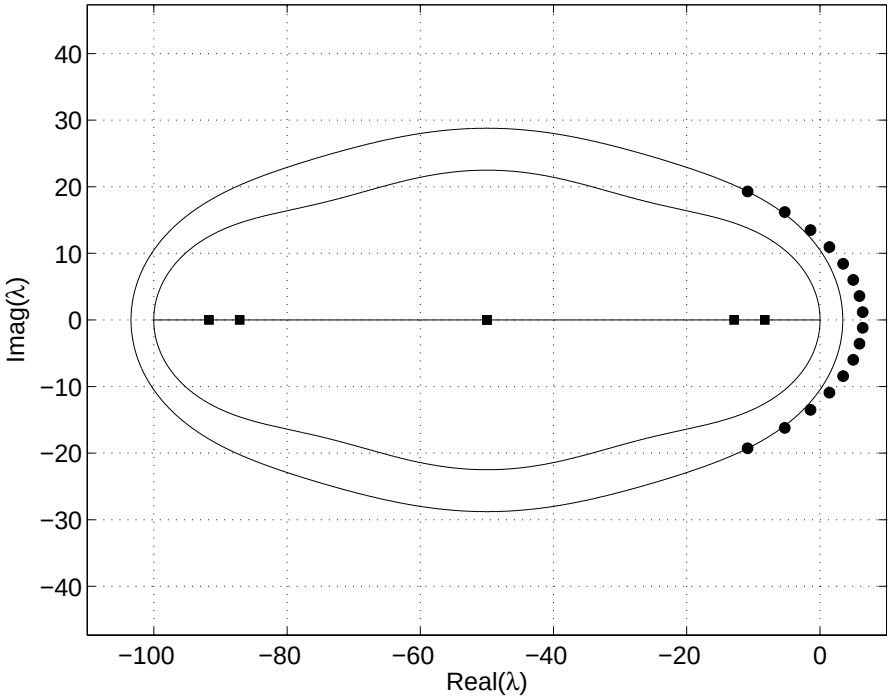
- more precisely,

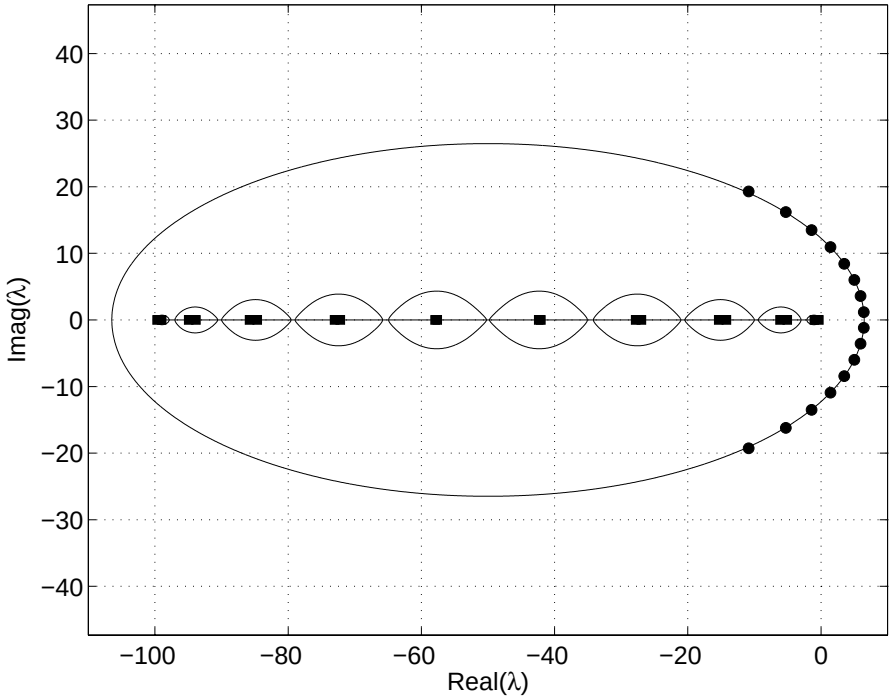
$$\limsup_{k\to\infty} \|\widehat{\boldsymbol{f}}_k - r_2(A)\boldsymbol{b}\|^{1/km} = \frac{\tau_A}{\tau_s},$$

where

$$\tau_A = \min\{\tau \,:\, \Lambda(A) \text{ is contained in the closed interior of } L_\tau\},$$

$$\tau_s = \max\{\tau \,:\, r_2 \text{ is analytic in the interior of } L_\tau\}.$$





To summarize: Algorithm 2 converges as follows

- initially, behavior like polynomial interpolation to $\exp$ (no error reduction in the startup phase and then superlinear convergence) as long as $\frac{d^{km} r_2}{d\lambda^{km}} \approx \frac{d^{km} \exp}{d\lambda^{km}}$,
- there is a point from where on the poles of r_2 become visible, fairly independent of the restart length,
- after this point of transition, behavior like polynomial interpolation to r_2 (slower linear convergence or even linear divergence).

The aim is to choose the restart length large enough such that at the point of transition the desired accuracy is reached.

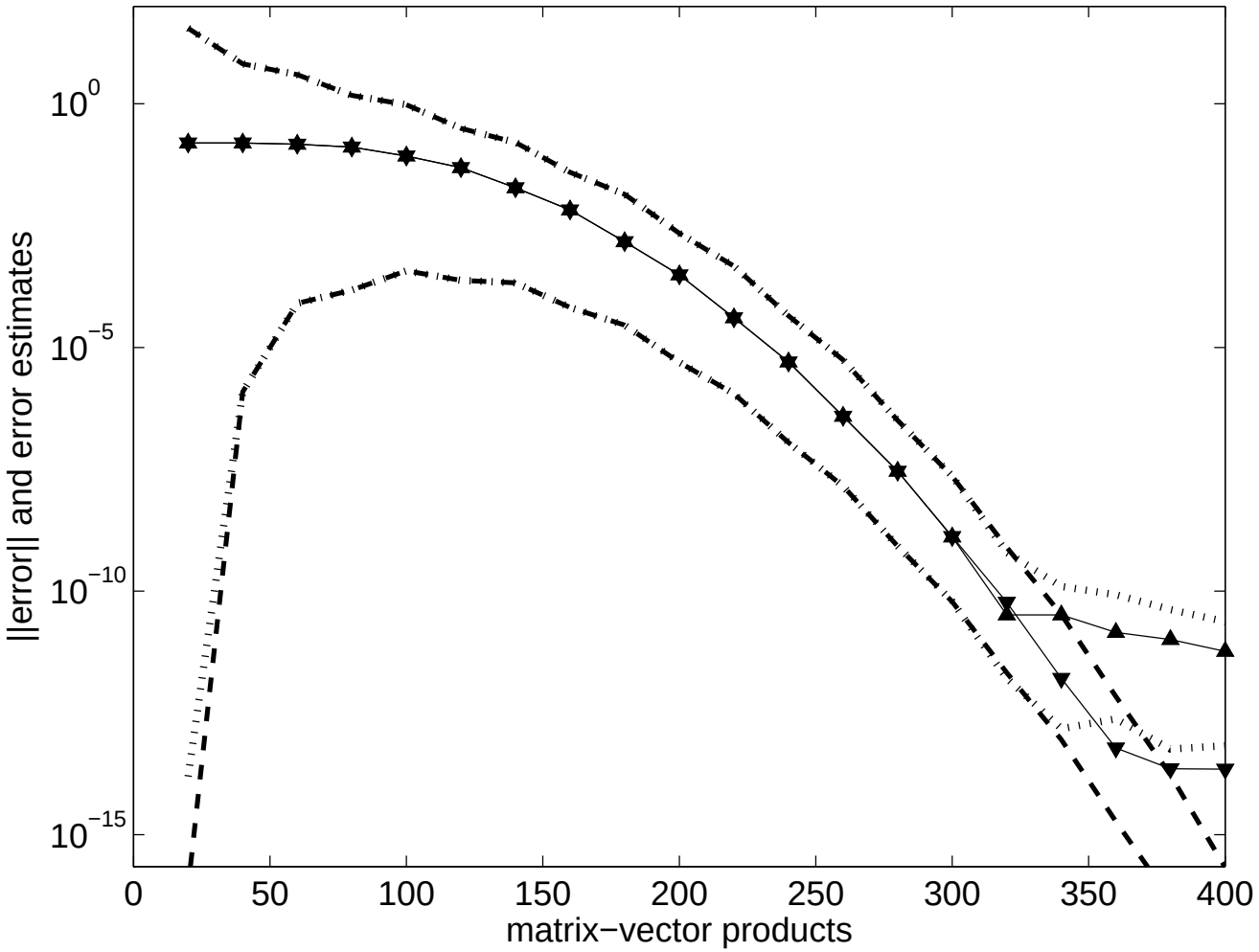
Stopping criteria (very shortly)

Recall: $\widehat{\boldsymbol{f}}_k = p_{km-1}(A)\boldsymbol{b}$ where p_{km-1} interpolates f at km eigenvalues of $\widehat{H}_k$.

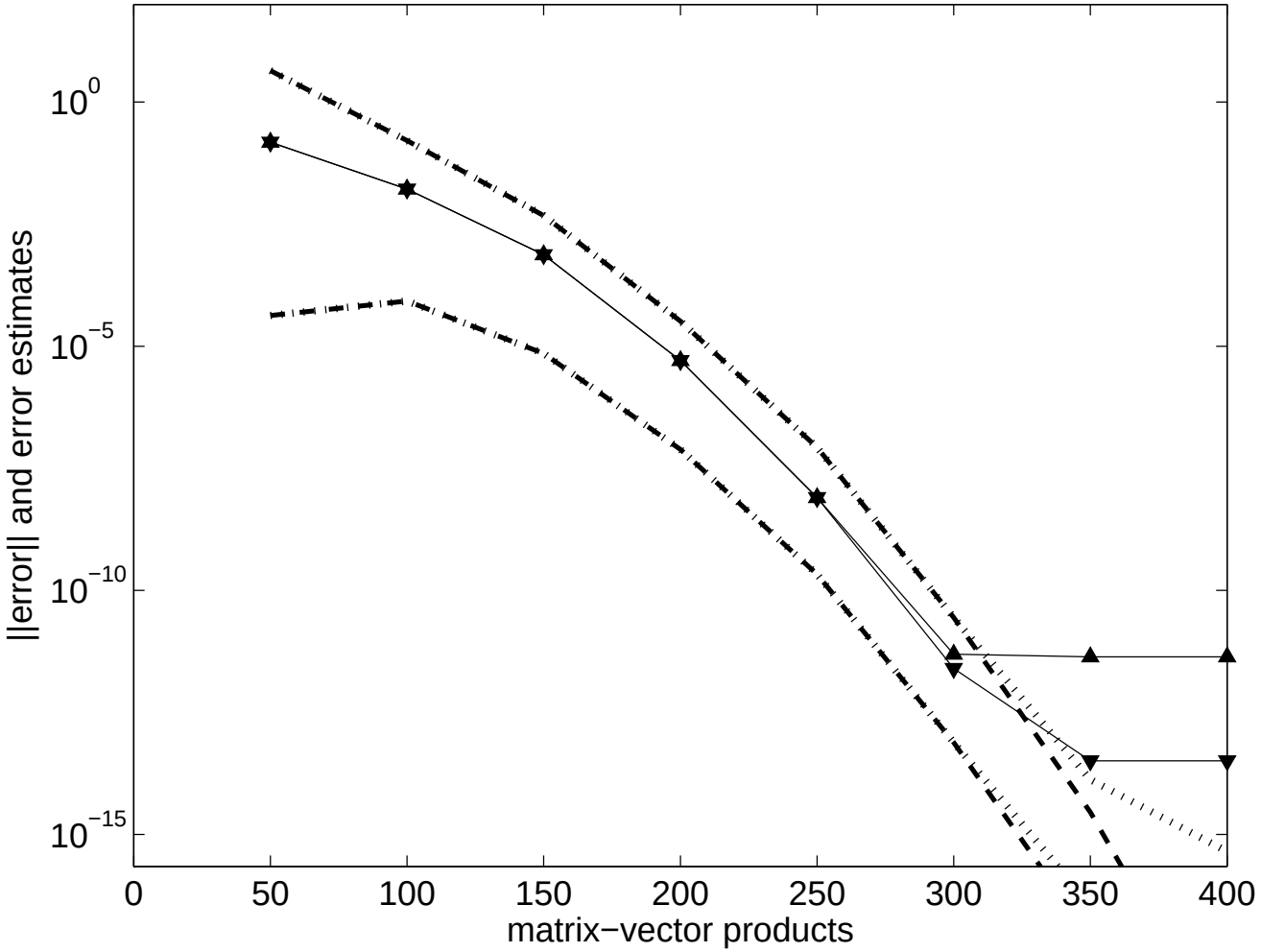
Idea:

- Generalize 'extended scheme' of [Saad, 1992] or [Phillipe & Sidje, 1993] by adding $\widetilde{m}$ interpolation nodes $\vartheta_1, \vartheta_2, \ldots, \vartheta_{\widetilde{m}}$.
- Close connection Lanczos $\Longleftrightarrow$ OP $\Longleftrightarrow$ Quadrature.
- Special case $\widetilde{m} = 1$: set $\vartheta_1 = \min \Lambda(A) \approx \min \Lambda(\widehat{H}_k)$
  $\Longrightarrow$ Gauss-Radau quadrature of $\exp$
  $\Longrightarrow$ lower bound.
- Special case $\widetilde{m} = 2$: set $\vartheta_2 = \max \Lambda(A) \approx \max \Lambda(\widehat{H}_k)$
  $\Longrightarrow$ Gauss-Lobatto quadrature of $\exp$
  $\Longrightarrow$ upper bound.

3D Heat equation, $n = 50$, $f = \exp$, $A \in \mathbb{R}^{125{,}000 \times 125{,}000}$, restart length $m = 20$.



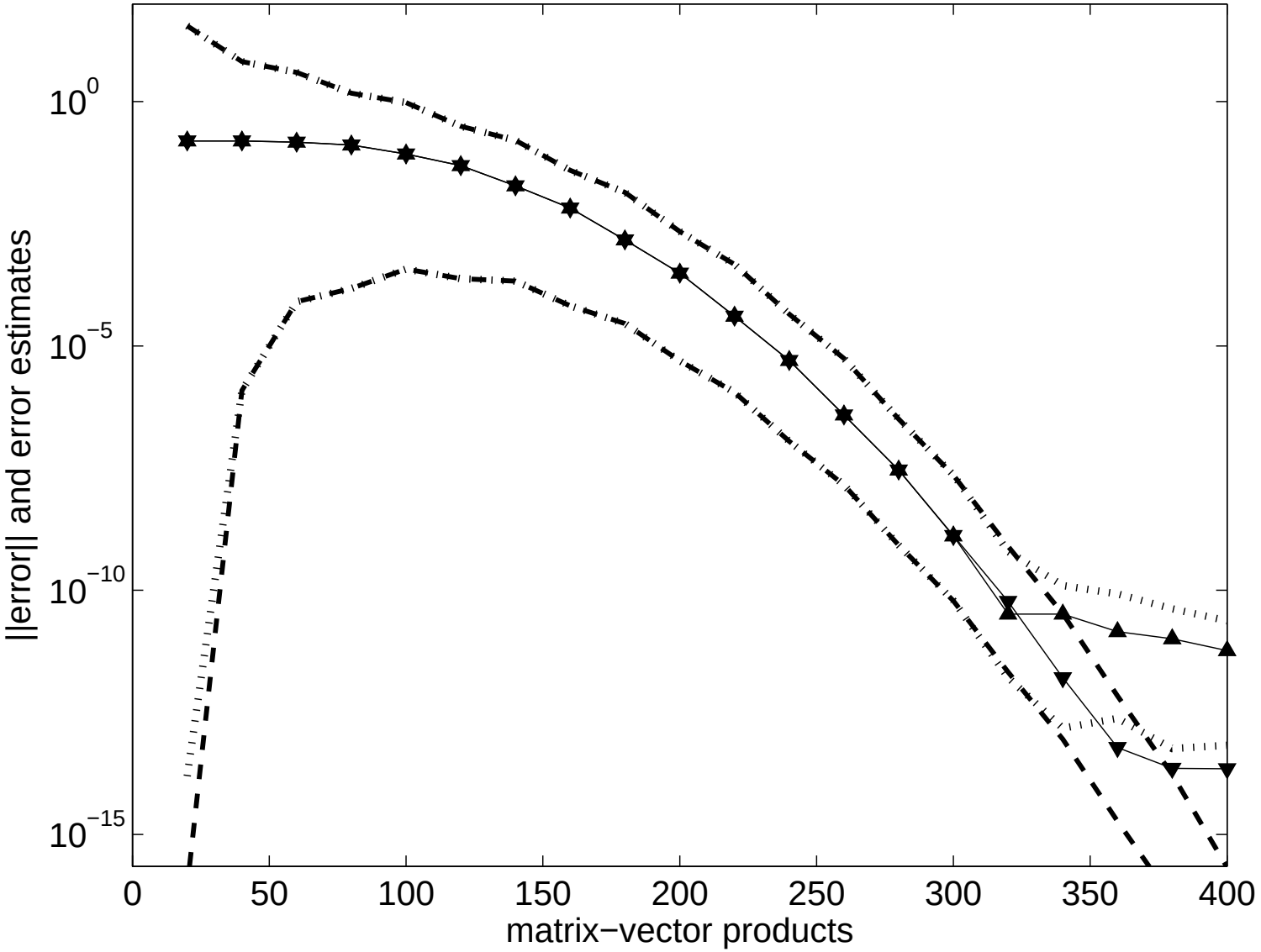
3D Heat equation, $n = 50$, $f = \exp$, $A \in \mathbb{R}^{125{,}000 \times 125{,}000}$, restart length $m = 50$.

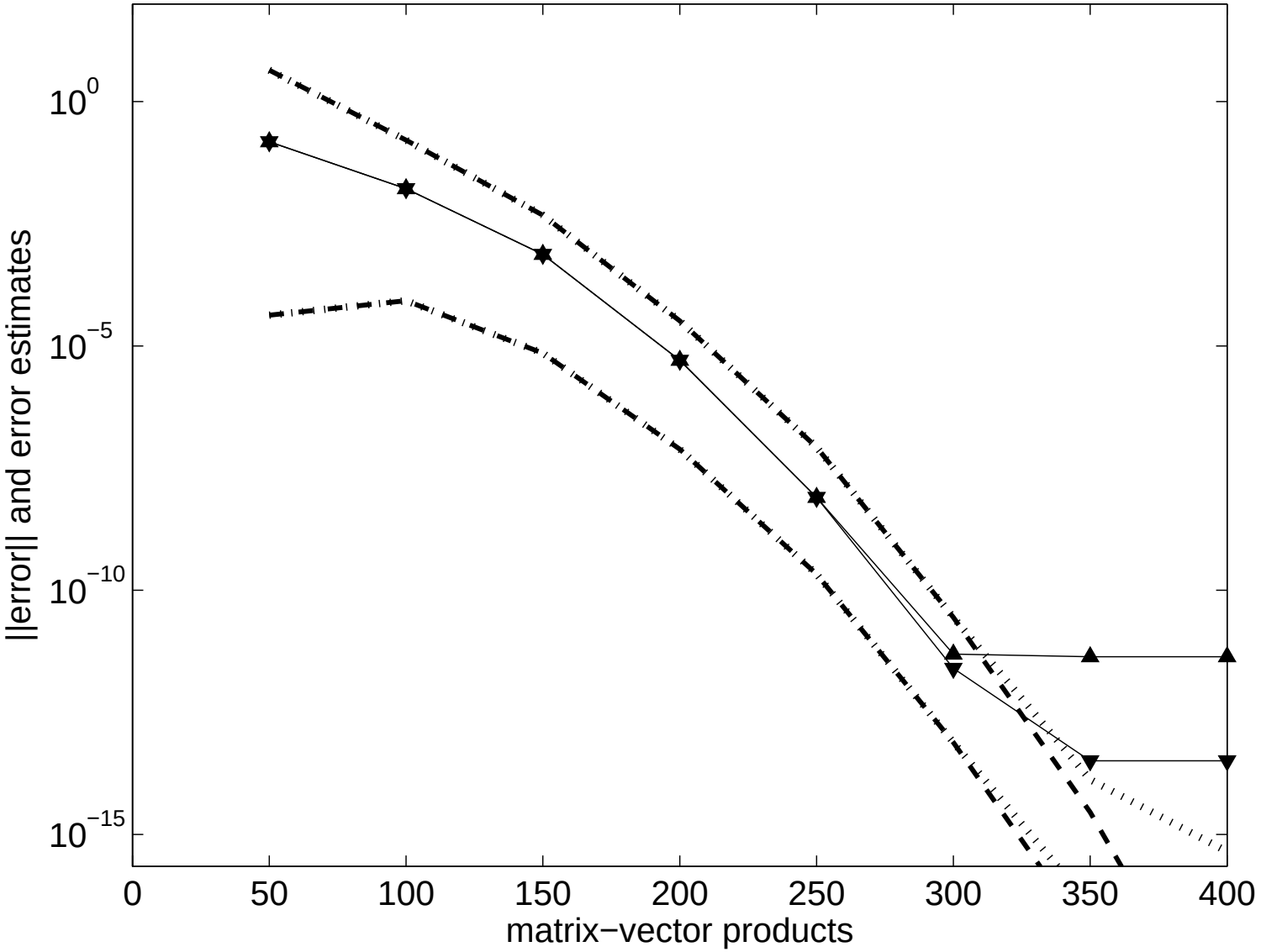


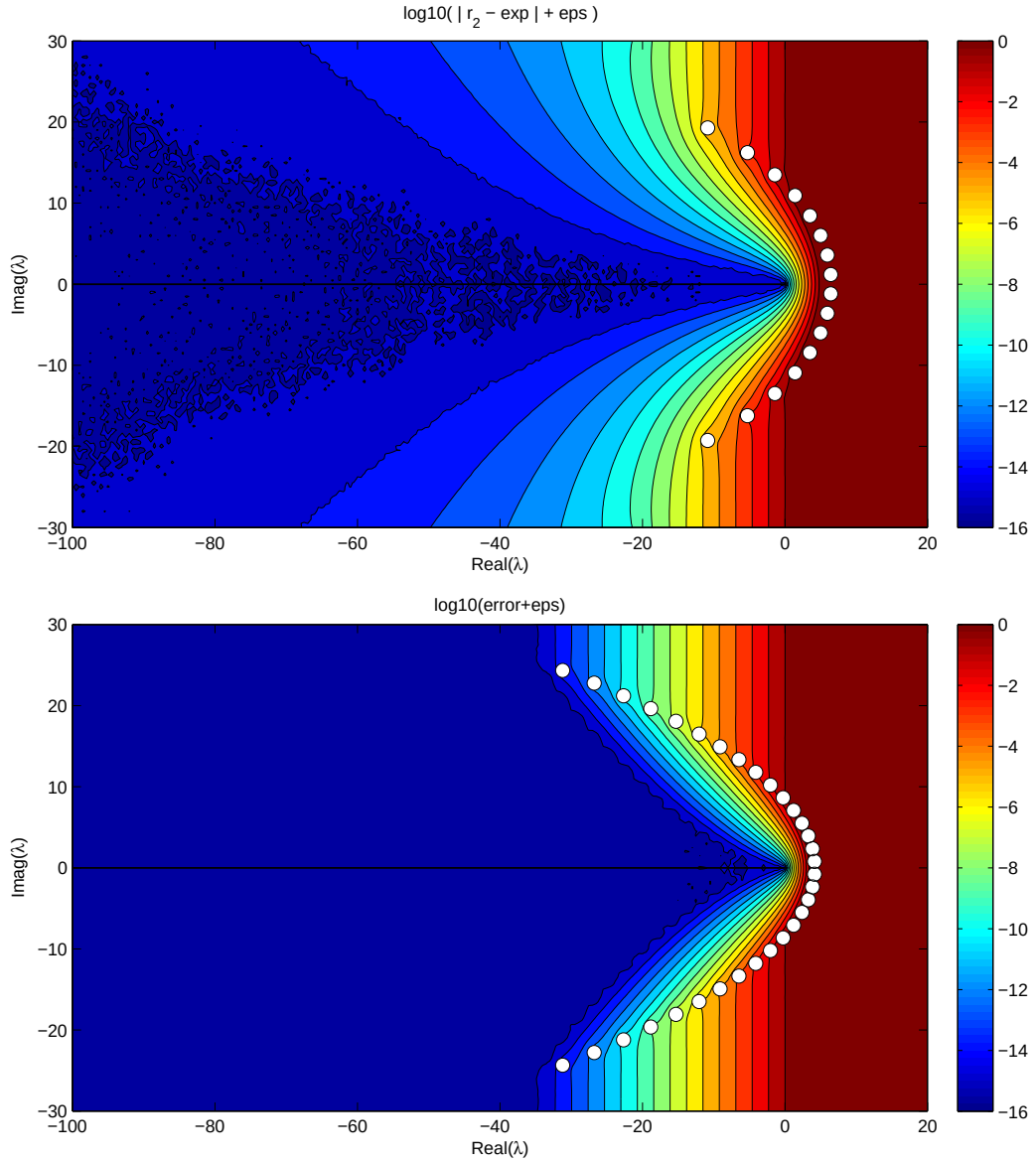
Summary

- Restarted Arnoldi methods result in acceptable storage cost even for very large matrices.
- The new implementation (Algorithm 2) is faster as it solves only a fixed number of linear equations of the size of the restart length m instead of, as is the case with Algorithm 1, evaluating a function of a matrix of increasing size km in the k-th cycle.
- The method converges if the restart length is sufficiently large. The asymptotic convergence behavior is (nearly) understood.
- Stopping criteria are available.
- **To do:** Spectral transformation, augmenting, preconditioning, rational approximations suited for other classes of functions and/or matrices (as, e.g., in [Trefethen, Weideman & Schmelzer, 2006]).

Thank you!


10^{0}
10^{-5}
10^{-10}
10^{-15}
||error|| and error estimates
0
50
100
150
200
250
300
350
400
matrix−vector products


10^{0}
10^{-5}
10^{-10}
10^{-15}
||error|| and error estimates
0
50
100
150
200
250
300
350
400
matrix−vector products


log10(| r2 − exp | + eps)
Imag(λ)
Real(λ)
log10(error+eps)
Imag(λ)
Real(λ)

An Advection-Diffusion Problem

$$
\begin{aligned}
\partial_t u - \frac{1}{Pe}\Delta u + \boldsymbol{a}\cdot\nabla u &= 0 && \text{in } \Omega = (-1,1)\times(0,1),\\
u &= 1-\tanh(Pe) && \text{on } \Gamma_0,\\
u &= 1+\tanh((2x+1)Pe), && \text{on } \Gamma_{\mathsf{in}},\\
\frac{\partial u}{\partial n} &= 0 && \text{on } \Gamma_{\mathsf{out}}.
\end{aligned}
$$

$$
\boldsymbol{a}(x,y) = \Big[2y(1-x^2),\, -2x(1-y^2)\Big], \qquad (x,y)\in\Omega
$$