Asignment I

SF2524, Group52

Jiacheng Xu jiacxu@kth.se Xinyi Ji jxinyi@kth.se

November 24, 2025

1

AI usage level: 2

2

We compute the eigenvalues, with an error tolerance of $\epsilon < 10^{-10}$

(a) Power Method

Convergence Rate:

The power method converges to the largest eigenvalue $\lambda_1 \approx 4.88$ with linear convergence. The convergence rate is directly related to the ratio $|\lambda_2/\lambda_1| \approx 0.54$, where $\lambda_2 \approx 2.65$ is the second-largest eigenvalue.

Analysis:

According to convergence of the power method (page 4), the error decreases proportionally to $(|\lambda_2/\lambda_1|)^k$ at iteration k. Since $|\lambda_2/\lambda_1| \approx 0.54$, the error reduces by approximately 57.5% per iteration, which is consistent with the observed linear convergence in the semilog plot.

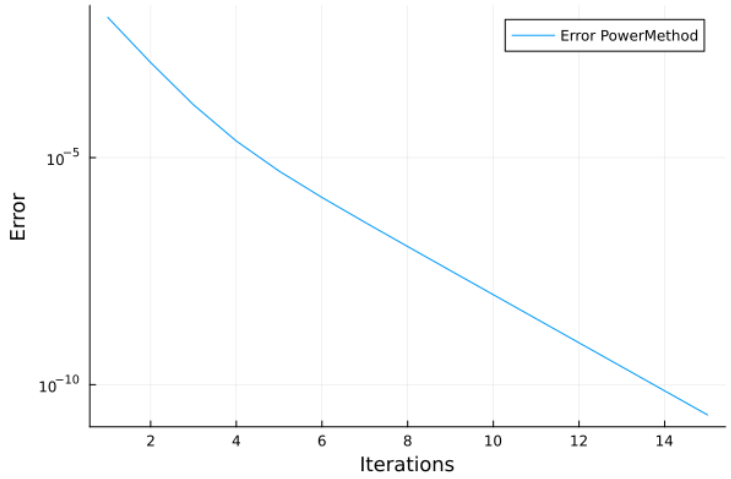


Figure 1: Power Method

(b) Inverse Iteration with Fixed Shift $\mu = 3$

Converges to:

The inverse iteration with shift $\mu = 3$ converges to the eigenvalue $\lambda_2 \approx 3.0$, which is the eigenvalue closest to the shift value.

Convergence Rate Comparison:

Inverse iteration converges faster than the power method. The convergence rate is proportional to the ratio of distances from the shift to the eigenvalues. Since $\mu = 3$ is very close to $\lambda_2 \approx 3.0$. Reference page 5.

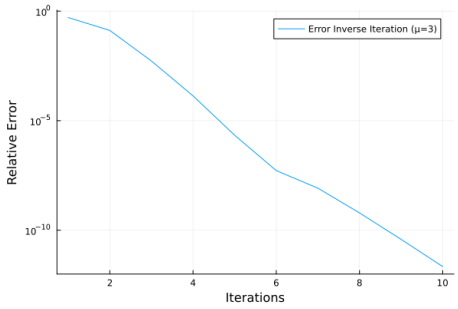


Figure 2: InverseIteration Shift

(c) Rayleigh Quotient Iteration

Observed Convergence Rate:

Rayleigh quotient iteration (RQI) exhibits cubic convergence. The error decreases rapidly, converging to machine precision in just 3 iterations.

Explanation:

The cubic convergence reference page 6.

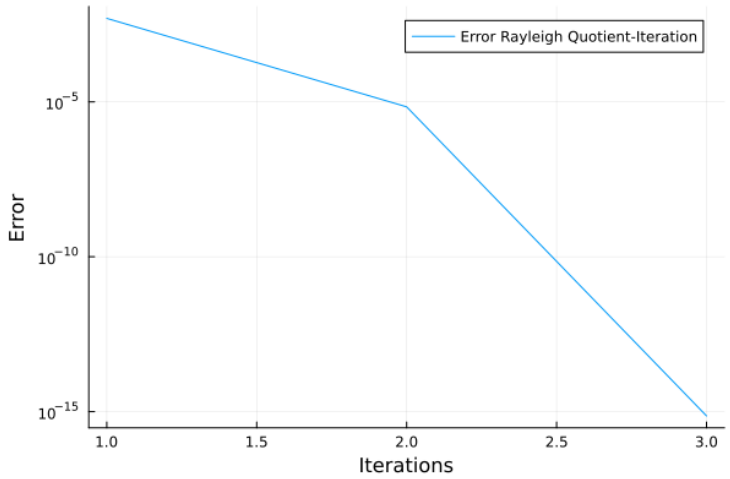


Figure 3: Rayleigh Quotient Iteration original

(d) Perturbed Matrix ($b_{2,1} = 1.5$)

Perturbed Matrix:

$$B_{\text{perturbed}} = \begin{bmatrix} 4 & 1 & 1 \\ 1.5 & 3 & 0 \\ 1 & 0 & 2 \end{bmatrix}$$

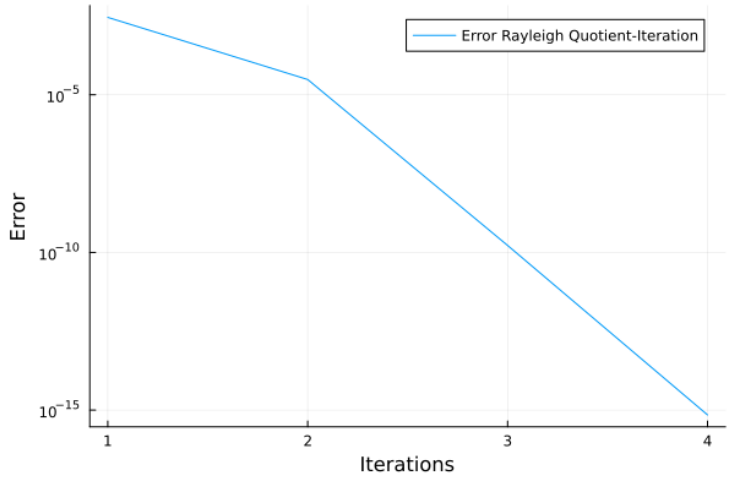


Figure 4: Rayleigh Quotient Iteration perturbed

From 3 iteration to 4 iteration.

Convergence is Slower because The original matrix B is symmetric, but the perturbed matrix is not symmetric. cubic convergence rate of RQI is downgrade to double. reference page 6.

3

Using table generation function from Julia package Latexify.

m	Time (MV)	Orth (MV)	Time (for)	Orth (for)	Time (MGS)	Orth (MGS)
5	0.00	5.06e-14	0.00	3.65e-14	0.00	3.65e-14
10	0.00	2.88e-13	0.01	1.54e-13	0.00	1.54e-13
20	0.01	1.30e-12	0.02	5.77e-13	0.01	5.77e-13
50	0.02	1.30e-11	0.05	3.63e-12	0.05	3.63e-12
100	0.05	7.86e-10	0.15	1.14e-10	0.25	1.14e-10

Table 1: Implementation Comparison

- **Speed:** The Classical Gram-Schmidt (CGS) using matrix-vector products (MV version) is significantly faster than the versions using for-loops. This is because the MV version leverages better cache locality.
- **Orthogonality:** The CGS (MV) version suffers from the worst loss of orthogonality, with an average error of $\approx 1.6 \times 10^{-10}$. The Modified Gram-Schmidt (MGS) and CGS (for-loop) versions perform identically, achieving much better orthogonality ($\approx 2.4 \times 10^{-11}$).

m	Time (1×)	Orth (1×)	Time (2×)	Orth (2×)	Time (3×)	Orth (3×)
5	0.00	5.06e-14	0.00	4.98e-15	0.00	5.08e-15
10	0.00	2.88e-13	0.00	7.21e-15	0.00	7.35e-15
20	0.01	1.30e-12	0.01	1.08e-14	0.01	1.09e-14
50	0.02	1.30e-11	0.02	1.92e-14	0.02	1.78e-14
100	0.04	7.86e-10	0.05	2.66e-14	0.06	2.60e-14

Table 2: Re-orthogonalization

This table demonstrates the effectiveness of re-orthogonalization.

- Applying Gram-Schmidt a second time (Double GS) dramatically improves orthogonality, reducing the error from $\approx 1.6 \times 10^{-10}$ (Single GS) to $\approx 1.4 \times 10^{-14}$, which is near machine precision.
- The computational cost for this improvement is moderate, increasing the runtime by only about 25% on average (from ≈ 0.013s to ≈ 0.016s).
- A third pass (Triple GS) offers almost no additional orthogonality improvement over the double pass but further increases the cost.

Which version is "best"?

- **Best for Speed:** Single Classical GS (MV version). This is the fastest but least accurate.
- **Best for Accuracy:** Double Classical GS (MV version) or Modified GS. Double CGS (MV) achieves near-perfect orthogonality ($\approx 1.4 \times 10^{-14}$) and is still very fast.
- **Best Balance:** The **Double Classical GS (MV version)** provides the best balance of speed and numerical stability.

Fairness of Comparison

Measureing and comparing the computation time in Julia is not straight forward, as as Julia's meomry reuse function, JIT Compilation, Reduced Array Reallocation and Garbage collection methods result in reduced allocations for subsequent function calls which leads to a bias towards parameters tested further down the line.

4

(a)

$K_m(A, b) = \text{span}\{b, Ab, A^2b, \ldots, A^{m-1}b\}$.

Both K_m and the Arnoldi method work within this subspace, there exists an invertible $m \times m$ matrix S such that:

$$K_m = Q_m S.$$

Using $K_m = Q_m S$:

$$K_m^\top K_m = (Q_m S)^\top (Q_m S) = S^\top Q_m^\top Q_m S = S^\top S \quad (\text{since } Q_m^\top Q_m = I),$$
$$K_m^\top A K_m = (Q_m S)^\top A (Q_m S) = S^\top Q_m^\top A Q_m S.$$

$$\mu w = (K_m^\top K_m)^{-1} K_m^\top A K_m w.$$

$$\mu w = (S^\top S)^{-1} S^\top Q_m^\top A Q_m S w = S^{-1} Q_m^\top A Q_m S w.$$

Let $z = Sw$, so $w = S^{-1}z$. Then:

$$\mu S^{-1} z = S^{-1} Q_m^\top A Q_m z.$$

Multiply both sides by S:

$$\mu S S^{-1} z = Q_m^\top A Q_m z \implies \mu z = Q_m^\top A Q_m z.$$

Finally the projected eigenvalue problem:

$$Q_m^\top A Q_m z = \mu z.$$

(b)

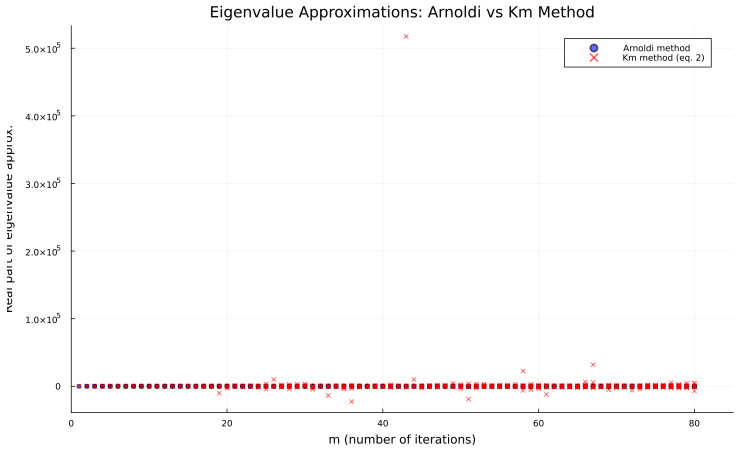


Figure 5: Compare Eigenvalue approx

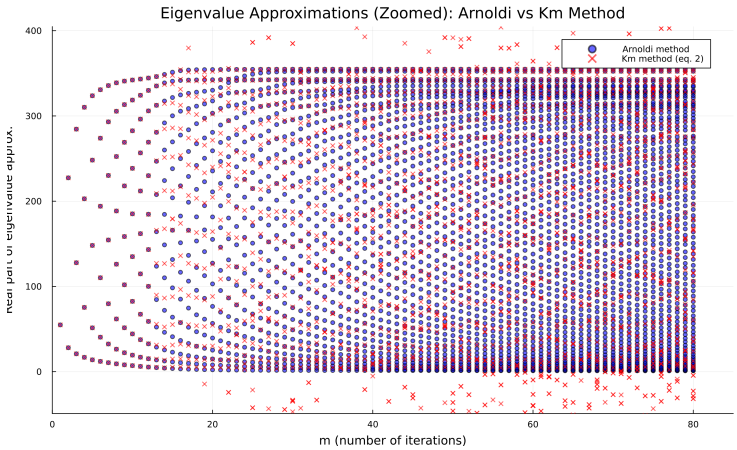


Figure 6: Zoomed Compare Eigenvalue approx

(c)

- **Exact arithmetic expectation.** Since both approaches use the same Krylov subspace $\mathcal{K}_m(A, b)$.

- **In finite precision.** For small m the two methods give similar approximations, but as m grows, columns of K_m become nearly linearly dependent, $K_m^\top K_m$ becomes ill-conditioned, and the $(K_m^\top K_m)^{-1} K_m^\top A K_m$ route develops large errors and spurious values.

- **Arnoldi is better**

5

(a) Figure 7 displays the eigenvalues of the Bwedge matrix. The eigenvalues for which we expect the fastest convergence with the Arnoldi method are highlighted and labeled by the letters A, B, and C. We expect a faster convergence rate for these eigenvalues due to their isolation from the remaining spectrum.

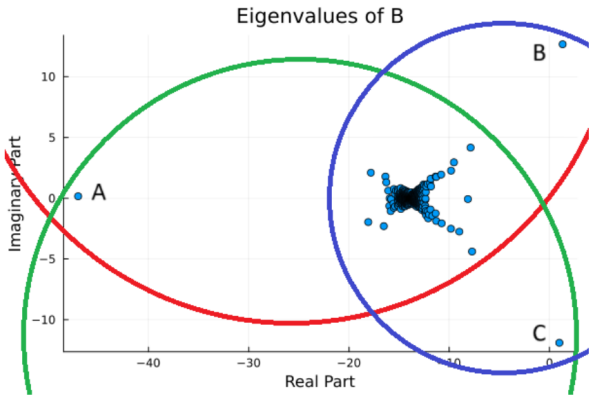


Figure 7: Eigenvalues of Bwedge. The three candidates with expected fast Arnoldi convergence are labeled A, B, and C.

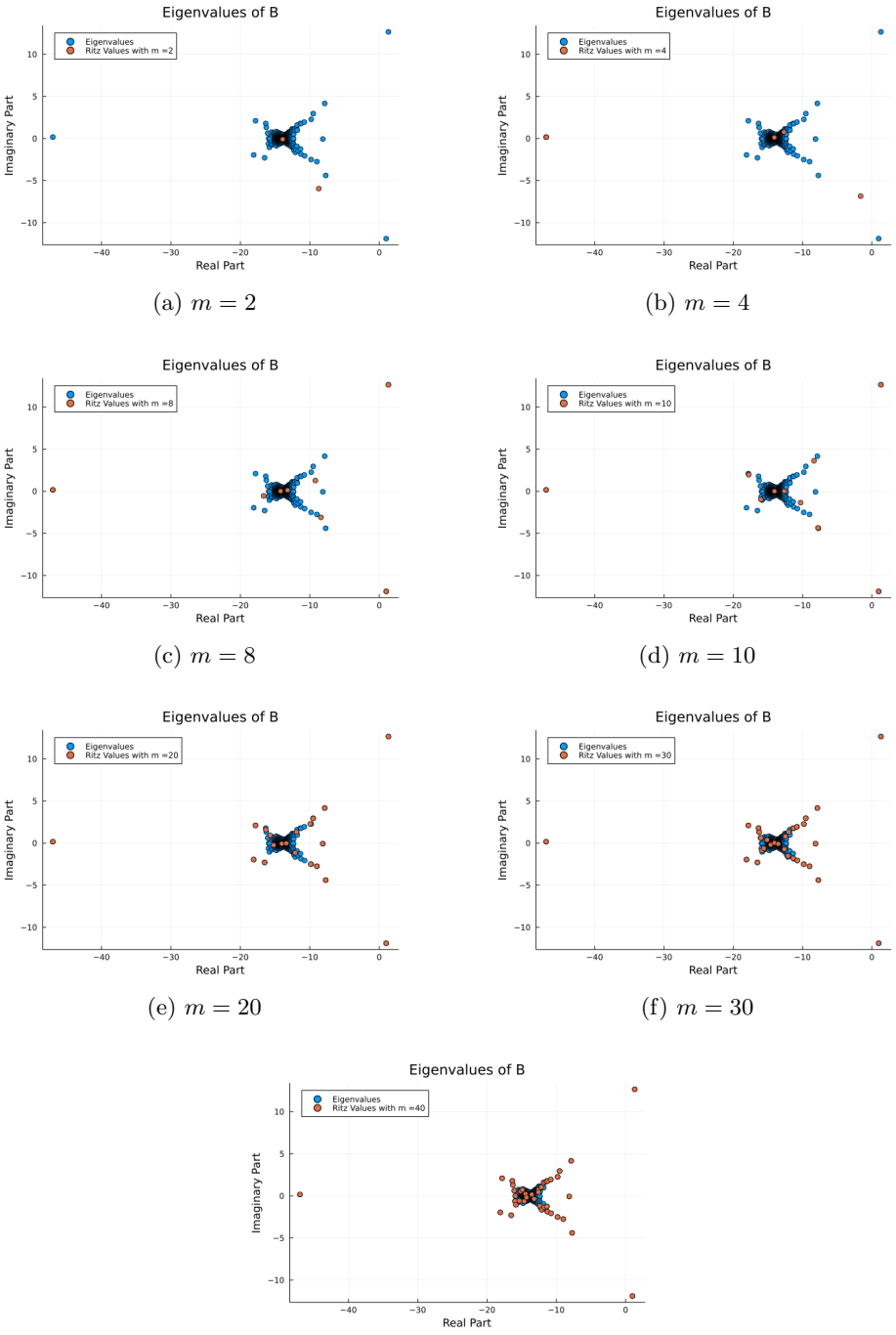
(b) In Figure 7 we also indicate three separation disks: the green disk includes all eigenvalues except B, the blue disk includes all eigenvalues except A, and the red disk includes all eigenvalues except C. The convergence factor is $|\lambda - c|$. Estimated convergence factor are, $A : 43, B : 29, C : 36.8$.

(c) We expect he fastest convergence towards the Eigenvalue A. At the iteration 4, we see the approximation of A.

To get eigenvalue approximate 10^{-10}, follow this table:

m	Best Approximated Eigenvalue	Min Relative Error
2	-13.9113 -0.0894i	3.9620e-04
4	-47.0161 +0.1659i	8.0165e-04
8	-47.0161 +0.1659i	1.0581e-06
10	-47.0161 +0.1659i	5.7083e-08
20	-47.0161 +0.1659i	2.2669e-15
30	0.9856 -11.8979i	1.0417e-15
40	0.9856 -11.8979i	1.3394e-15

(d)



(a) $m = 2$ (b) $m = 4$

(c) $m = 8$ (d) $m = 10$

(e) $m = 20$ (f) $m = 30$

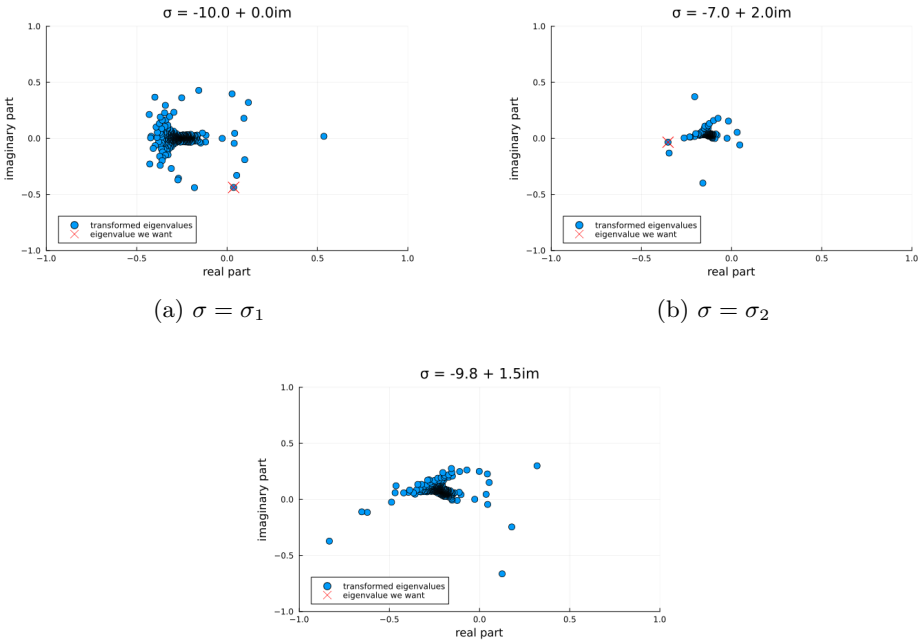
(g) $m = 40$

Figure 8: Ritz values plots for different iterates of m in the Arnoldi process

(e)

Method	m = 10	m = 20	m = 30
Standard AM	2.5031e+00	1.8574e-01	1.6756e-04
σ_1 = -10	1.0788e+00	1.8518e+00	1.8738e+00
σ_2 = -7 + 2i	2.6118e+00	2.4975e+00	2.4788e+00
σ_3 = -9.8 + 1.5i	2.6487e-01	5.0802e-01	5.1132e-01

- When σ is close to λ, the transformed eigenvalue $1/(\lambda * -\sigma)$ has large



(a) $\sigma = \sigma_1$

(b) $\sigma = \sigma_2$

(c) $\sigma = \sigma_3$ eigenvalue we want is too large

Figure 9: transferred eigenvalues in shift-and-invert Arnoldi method

magnitude and is well-separated from other eigenvalues.

- $\sigma_3 = -9.8 + 1.5i$ is closest to $\lambda* \approx -9.8 + 2i$,hence shows best convergence performance.

6

(a)

This video introduces some application about large sparse matrix. Like Laplacian matrix, every data point relate to a node in the graph, normally nodes are unrelated.

(b)

The Rayleigh-Ritz method is a way to approximate eigenvalues and eigenvectors of a large matrix by reducing it from an $n \times n$ size to a smaller $k \times k$ matrix. It find solution in the search subspace $span(q_1 ... q_n)$ that eigenvector $x \approx Qz$. If the subspace contains an eigenvector, the method provides exact results.

(c)

Dividing by beta (zero vaule in Hessenberg matrix) in the Gram-Schmidt process within the Arnoldi method causes the breakdown. A zero in the last entry of the Hessenberg matrix results in a zero residual, indicating an exact eigenvalue.

(d)

The shift-and-invert Arnoldi method apples the Arnoldi process to the matrix $A - \sigma I^{-1}$. While the classical Arnoldi method converges quickly to isolated and extreme eigenvalues, the shift-and-invert technique targets eigenvalues near the shift σ. We can use matrix decomposition (like LU or Cholesky) to simplify inversion or solve the linear system that appears in each Arnoldi iteration.

8

(b) The size of matrix X_c is (4096 * 400). The size of matrix C is (4096 * 4096). The computation times are measured and shown in Table 1

Process	(1)	(2)	(3)
Time(sec)	0.233	0.433	0.041

Table 1: Comparison of computation times

(c) Power Method based on **(b.3)** is faster, as shown in Table 2. The eigenface corresponding to the largest eigenvalue is shown in Fig. 10.

based matrix	C	X_c
Time(sec)	3.64	0.02

Table 2: Comparison of computation times of Power Method

Figure 10: Eigenface corresponding to λ_1

(d) The convergence plot of Arnoldi Method is shown in Fig. 11. The eigenfaces corresponding to five largest eigenvalues are shown in Fig. 12.

(e)

- Power Method only converges to the biggest eigenvalue and the corresponding eigenvector. The convergence depends on $|\frac{\lambda_2}{\lambda_1}|$. The smaller the value is, the slower it converges.

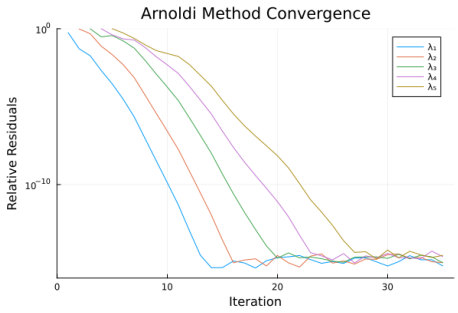


Figure 11: The convergence of Arnoldi

(a) Eigenface 1 (b) Eigenface 2 (c) Eigenface 3

(d) Eigenface 4 (e) Eigenface 5

Figure 12: First 5 eigenfaces

- Arnoldi Method can converge to several biggest eigenvalues and the corresponding eigenvectors. It converges faster than Power Method.
- Arnoldi method is preferable because it's more numerically stable and provides multiple eigenvalues/eigenvectors simultaneously.

(f) The eigenface corresponding of λ_6 and the negative version of it are shown in Fig. 13.

(a) Eigenface 6

(b) Eigenface_Neg 6

Figure 13: Comparison of eigenface and its negative version.