Krylov Subspace Methods

Fast Iterative Solvers, Project 1

Johannes Leonard Grafen

June 5, 2023

Contents

List of Figures

List of Tables

1 Remarks on used architecture and compiler options

The code was compiled using Clang-compiler and ran on an Apple Silicon M1 Pro Chip with ARM64 architecture. All timings were conducted with the a CPU-timer object of the Boost timer library, which was specifically compiled for the previously mentioned CPU architecture. If not stated otherwise, no optimization using -O0 flag was used. For timings, the output was disabled at specific regions of the code, using a specific preprocessor directive "DISABLEIO" which can be passed to the compiler using -D option.

2 Generalized Minimal Residual Method - GMRES

2.1 Relative residuals for GMRES with and without preconditioning

For the full GMRES method, the relative residuals are plotted against the iterations index of the top loop in the full GMRES algorithm in Fig. 1. The iterations index in this case corresponds to the Krylov vectors. The number of necessary Krylov vectors to reduce the 2-norm of the initial residual by 8 orders of magnitude ($||\mathbf{r}_k||/||\mathbf{r}_0|| = 10^{-8}$) is given by $\tilde{m}$ in the plot's legend. All iterations were started with parameter $m = 600$ in case the convergence criterion was fulfilled before reaching the prescribed max number of Krylov vectors, the loop was exited prematurely with $\tilde{m}$. Compared to the full GMRES method without preconditioning ($\tilde{m} = 479$) the Gauss-Seidel preconditioning performed best with just $\tilde{m} = 143$ Krylov vectors that are required to established convergence (according to the aforementioned convergence criterion). I obtained $\tilde{m} = 256$ required Krylov vectors for Jacobi preconditioning and $\tilde{m} = 465$ Krylov vectors for Incomplete LU Factorization (ILU(0)) to reach convergence.

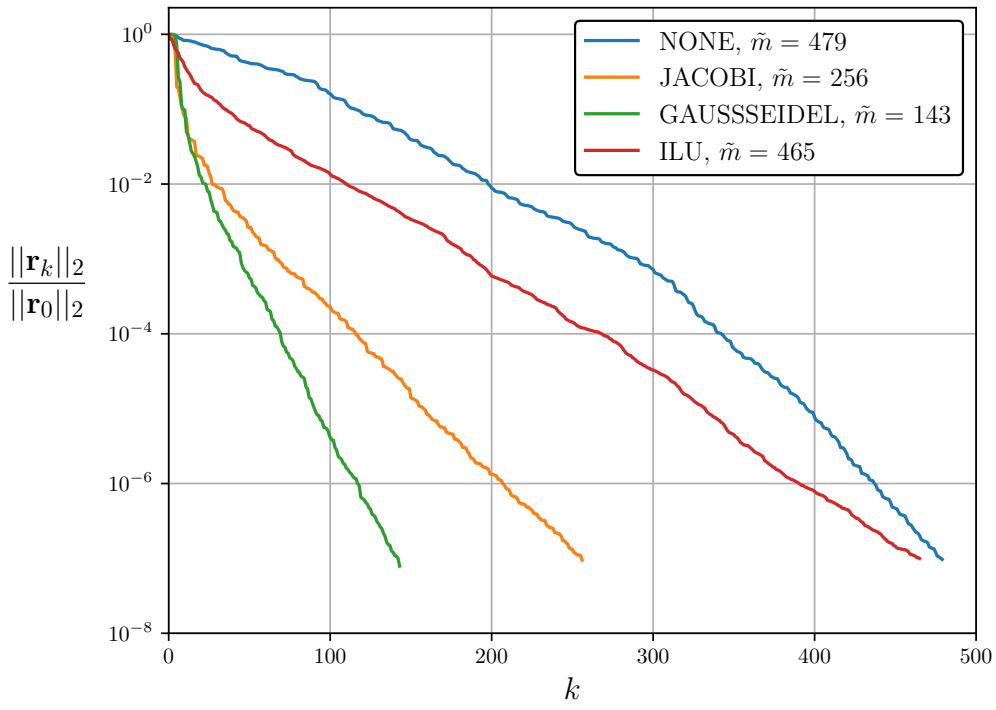

Figure 1: Comparison of the different preconditioning options for the GMRES procedure to the full GMRES Method without preconditioning using m = 600 Krylov vectors as an input.

2.2 Optimization of restart parameter for restarted GMRES

In an effort to find a "good" restart parameter the parameters $m = 30, m = 50$ and $m = 100$ are compared to the runtime of full GMRES in table Tab. 1 for no optimization and in table Tab. 2 for moderate optimization level. Restart parameters of $m = 50$ and $m = 100$ showed a decrease in

m	Iterations	User $[s]$	System $[s]$	CPU-Time $[s]$
479 (full)	1	5.04	0.11	5.15
30	144	7.25	0.08	7.33
50	43	4.43	0.06	4.49
100	14	4.23	0.09	4.32

Table 1: Comparison of timings for different restart parameters using clang compiler with no optimization (-O0 flag) + disabled Output (-DDISABLEIO) and without preconditioning

m	Iterations	User $[s]$	System $[s]$	CPU-Time $[s]$
479 (full)	1	0.29	0.1	0.39
30	143	0.46	0.06	0.52
50	43	0.29	0.07	0.36
100	14	0.27	0.07	0.34

Table 2: Comparison of timings for different restart parameters using clang compiler with optimization (-O2 flag) + disabled Output (-DDISABLEIO) and without preconditioning

CPU-Time of 12.8% and 16.1% for restarted GMRES compared to full GMRES [1] respectively, where as a restart parameter of $m = 30$ showed an increase of 43%, if compiler optimization is disabled. Activating a moderate optimization level (-O2) restart parameters of $m = 50$ and $m = 100$ decreased the CPU-Time only by 7.7% and 12.82%, compared to full GMRES.

Furthermore, I conducted an investigation to find the globally "best" restart parameter, which is presented in Fig. 2. The globally best restart parameter was found to be $m = 63$, requiring 25 iterations to converge in 3.73 seconds, which is a decrease by 27.6% opposed to the full GMRES method. The red line in Fig. 2 denotes the CPU-time of the full GMRES method ($m = 479$). For all restart parameter m for which the blue line is below the CPU time for full GMRES (indicated by the red line), the restarted formulation of the GMRES method is faster than full GMRES. The

[1] in the following context "full" GMRES refers to a restarted GMRES method were only one loop iteration of the restarted GMRES method is necessary to establish convergence according to the given convergence criterion.

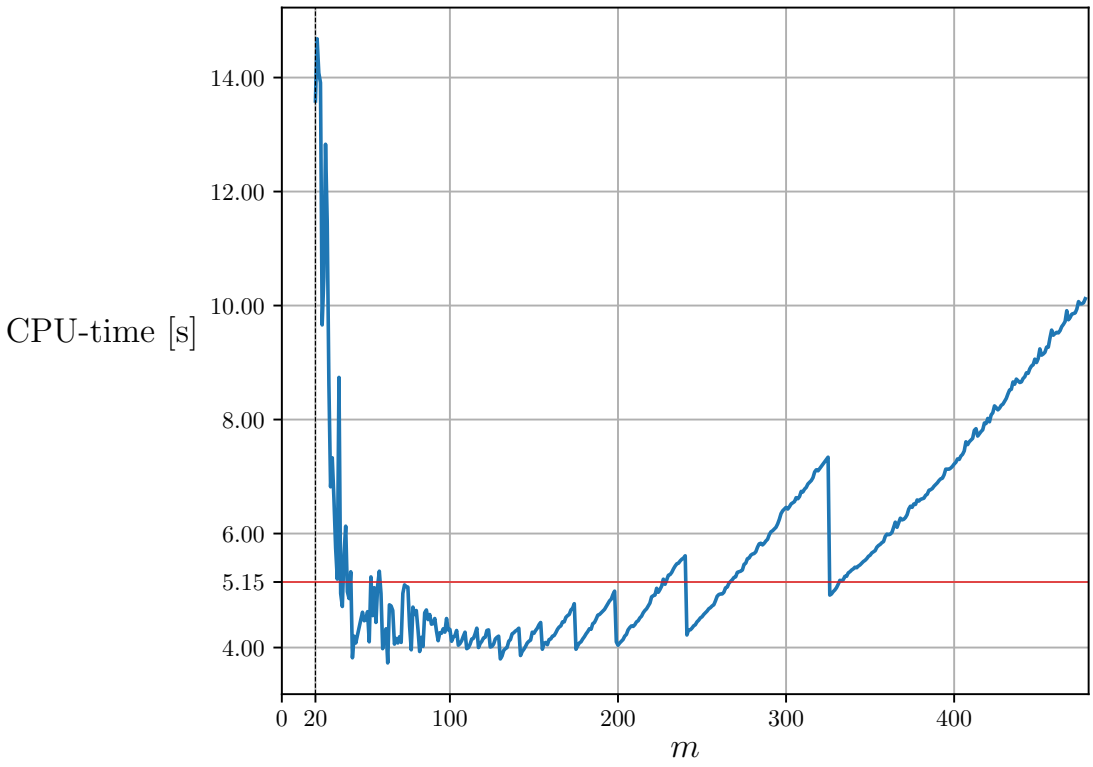

Figure 2: CPU-timings of restarted GMRES for different restart parameters $m = 20...478$ with global minimum at $m = 63$ with $3.73s$, using clang compiler with no optimization (-O0 flag) + disabled Output (-DDISABLEIO) and without preconditioning

steps in the course of CPU time can be attributed to the reduction of the required loop iterations. For example $m = 325$ require 3 iterations to converges opposed to $m = 326$ which only needs 2 iterations to converge. Due to an increased number of Krylov vectors m, the runtime of the top loop in the GMRES method increases as well. Besides a decreased runtime, the restarted formulation also allows for a decrease in required memory, as the stored Hessenberg matrix and also the matrix V that contains the Krylov vectors are smaller in the restarted formulation compared to the full GMRES method. If the input matrix $\mathbf{A}$ is very large, might require an unfeasible amount of memory.

2.3 Orthogonality of Krylov vectors

The orthogonality property of the Krylov vectors is represented by Fig. 3 for the full GMRES method. With an increasing number of Krylov vectors, the value of the dot products deviates more from zero, which means that the angle between the k-th Krylov vector $\mathbf{v}_k$ and the first Krylov vector $\mathbf{v}_1 = \mathbf{r}_0/||\mathbf{r}_0||_2$, deviates more from orthogonality. This is due to numerical error, that accumulates for increasing number of loop iterations (Krylov vectors) in the getKrylov function, that determines a new Krylov vector in each iteration step of the top loop in the full GMRES method.

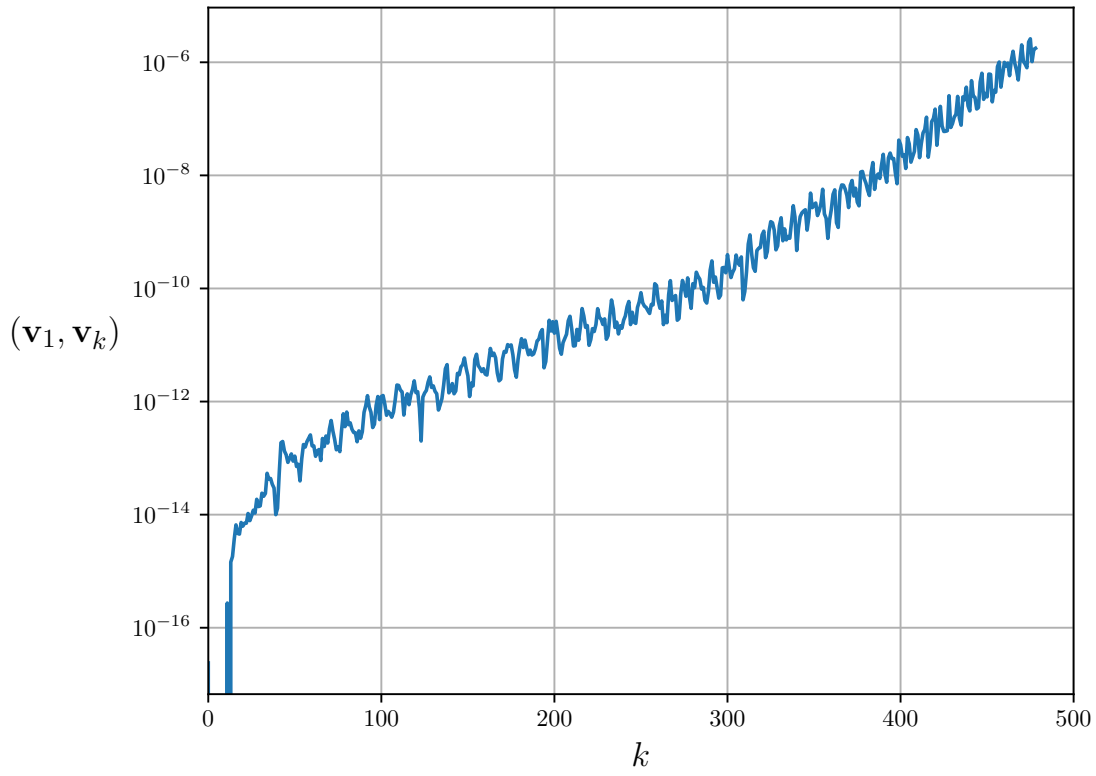

Figure 3: Orthogonality of the Krylov vectors. Plot of the dot product $(\mathbf{v}_1, \mathbf{v}_k)$ against the iteration index k of the top loop in the GMRES method

3 Conjugate Gradient Method - CG

For the given matrix the implemented CG method converges after $k = 21580$ iterations according to the same convergence criterion that was used for the GMRES method. The error $\mathbf{e}_k = \mathbf{x}_k - \mathbf{x}$ in A-norm $||\mathbf{e}_A|| = \sqrt{(\mathbf{Ae}, \mathbf{e})}$ as well as the residual $\mathbf{r} = \mathbf{Ax} - \mathbf{b}$ in 2-norm $||\mathbf{r}||_2 = \sqrt{(\mathbf{r}, \mathbf{r})}$ are plotted against the iteration index on a semi-og scale in Fig. 4. The known exact solution is denoted by $\mathbf{x}$. The graph of the error in A-norm of conjugate gradient method shows a smooth course were as the course of the residual is subject to strong fluctuations. To explain this behavior one should remind himself, that the CG method minimizes the error $\mathbf{e}$ along all search direction $\mathbf{p}_k$. In the lecture the "Optimality" was characterized as $(\mathbf{Ae}_k, \mathbf{p}_k)_A = 0, j = 0, ..., m-1$. The conjugate gradient method is formulated such that the error is always A-orthogonal to all previous search direction $\mathcal{P}_{m+1}$, meaning that the error in A norm is minimized in $\mathcal{P}_{m+1}$ according to best approximation theorem. For continuous iterations the error should thus be constantly minimized, approaching a global minimum. Where as the 2-Norm of the residual is not projected in search direction, which results in fluctuations. But the general trend of the residual is aimed to follow the one of the error in A-norm.

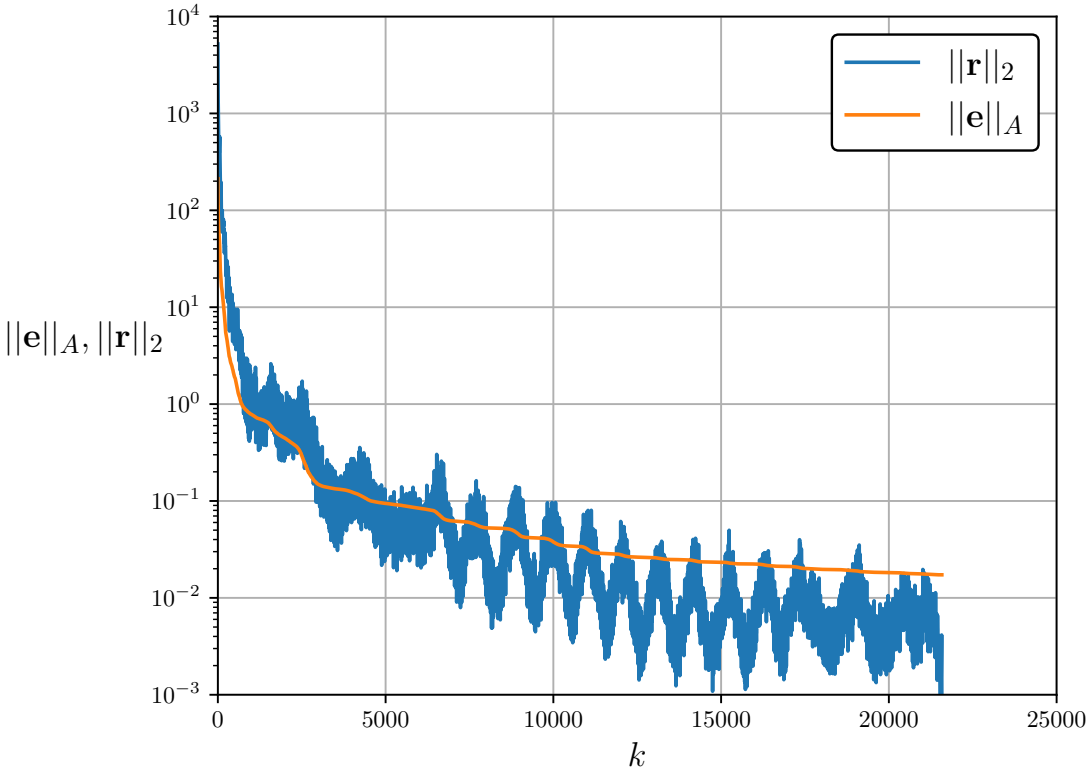

Figure 4: Error $\mathbf{e}_k = \mathbf{x}_k - \mathbf{x}$ in A-Norm $||\mathbf{e}||_A = \sqrt{(\mathbf{Ae}, \mathbf{e})}$ and the residual in 2-Norm $||\mathbf{r}||_2 = \sqrt{(\mathbf{r}, \mathbf{r})}$ against the iteration index k for CG-method