Exercise # 2. Iterative Methods For Linear Systems.

Alexandre Rodrigues (2039952)

January 7, 2022

Question 1

Using as a test the example usage, with $tol = 1 \times 10^{-8}$ and limiting the iterations to $maxit = 250$ I got the following results.

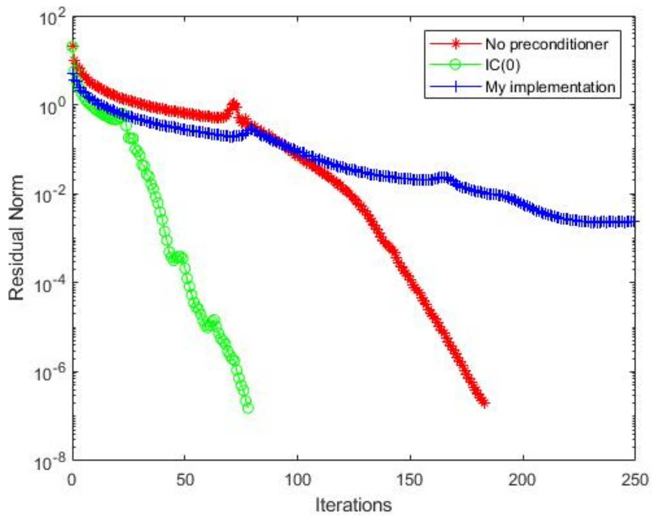

Figure 1: Residual norm vs iteration number for PCG methods, $maxit = 250$

Method	**Iterations**	**Final Residual**	**Computational Time**
Matlab PCG without preconditioning	183	1.9591×10^{-7}	$0.077s$
Matlab PCG IC(0)	78	1.5293×10^{-7}	$0.068s$
My PCG implementation	250	2.3×10^{-3}	$0.151s$

Table 1: Results of PCG methods, $maxit = 250$

When $maxit$ is large enough to guarantee convergence in all implementations we get the following results:

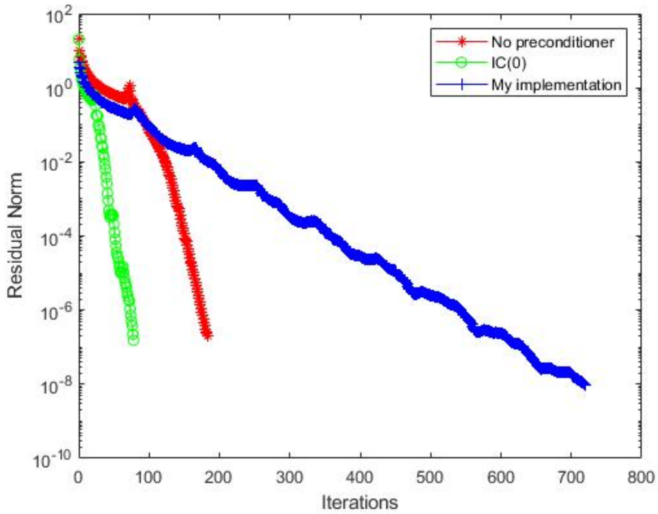

Figure 2: Residual norm vs iteration number for PCG methods, $maxit = 750$

Method	**Iterations**	**Final Residual**	**Computational Time**
Matlab PCG without preconditioning	183	1.9591×10^{-7}	$0.054s$
Matlab PCG IC(0)	78	1.5293×10^{-7}	$0.063s$
My PCG implementation	720	9.6833×10^{-9}	$0.351s$

Table 2: Results of PCG methods, $maxit = 750$

My implementation is slower to converge but produces better final residual values.

Question 2

The spectral condition number of A is

$$\kappa(A) = \frac{\lambda_{max}(A)}{\lambda_{min}(A)}. \tag{1}$$

In Matlab I used the `condest(A)` function to estimate the condition number of a sparse matrix A.

n_x	h	$\kappa(A)$	$\sqrt{\kappa(A)}$	CG	$PCG(0)$	$PCG(10^{-2})$	$PCG(10^{-3})$
102	1.0000×10^{-4}	6.0107×10^{3}	77.5288	283	87	45	17
202	2.5000×10^{-5}	2.3810×10^{4}	154.3039	532	159	78	30
402	6.2500×10^{-6}	9.4770×10^{4}	307.8473	948	282	137	53
802	1.5625×10^{-6}	3.7814×10^{5}	614.9304	1792	533	258	97

Table 3: Iterations of PCG methods for each value of n_x and respective values of h and $\kappa(A)$

One can note from the table the dependence of the number of iterations on $h = \frac{1}{N} = \frac{1}{(nx-2)^2}$. The number of iterations is halved when n_x approximately doubles.

Question 3

show theoretically ??

When using the Choledsky precontionier with no fill in, I did'nt get the expected results. Both Matlab's and my implementation converged in only one iteration.

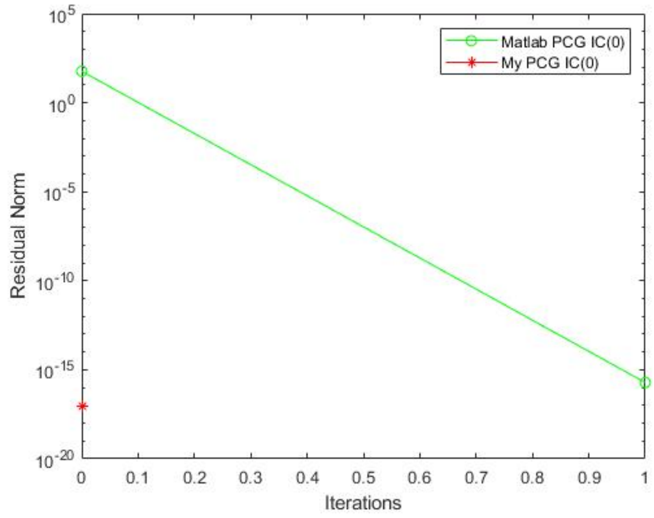

Figure 3: Residual norm vs iteration number for PCG methods with $IC(0)$ preconditioner

Due to the bad results, I tried to remove preconditioning form my implementation by setting L as the identity matrix, `L = speye(size(L))`.

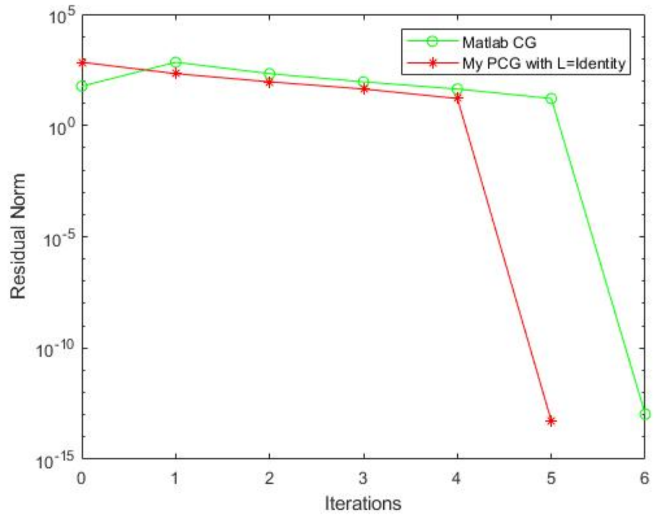

Figure 4: Residual norm vs iteration number for PCG methods without preconditioning

Method	Iterations	Final Residual	Computational Time
Matlab PCG	6	9.2128×10^{-14}	$0.021s$
My PCG	5	1.2744×10^{-13}	$0.012s$

Table 4: Results for each value of implementation, no preconditioning

These results show the theoretical calculations, my implementation is still better than expected.

Question 4

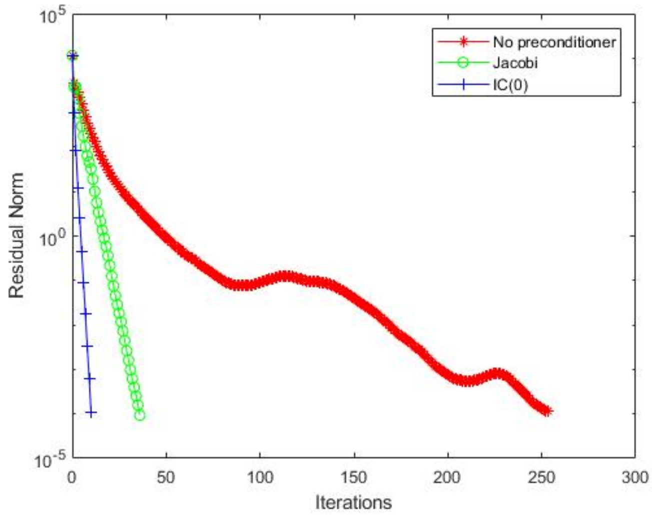

Figure 5: Residual norm vs iteration number for PCG methods without preconditioning

Preconditioner	**Iterations**	**Final Residual**	**Computational Time**
None	253	1.1367×10^{-4}	$0.254s$
Jacobi	36	9.3198×10^{-5}	$0.053s$
IC(0)	10	1.1155×10^{-4}	$0.046s$

Table 5: Results for each preconditioner

There is a very clear improvement when using preconditioning. It is also noticeable the superior characteristics of the incomplete Choledsky preconditioner relative to Jacobi.

Question 5

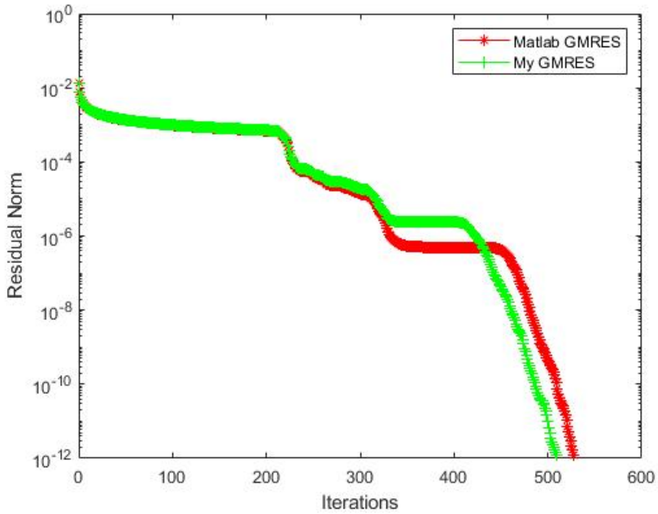

Figure 6: Residual norm vs iteration number for GMRES methods

Method	Iterations	Final Residual	Computational Time
Matlab GMRES	527	1.2073×10^{-12}	$9.097s$
My GMRES	509	1.2231×10^{-12}	$10.032s$

Table 6: Results for each GMRES implementation

These results show that the methods have very similar convergence characteristics. My implementation has a smaller number of iterations but the other results are slightly worse than the ones achieved with Matlab's implementation.

Question 6

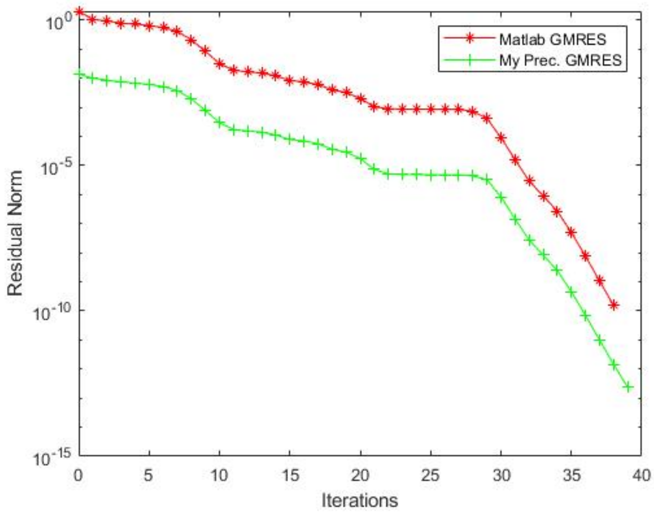

Figure 7: Residual norm vs iteration number for preconditioned GMRES methods

Method	Iterations	Final Residual	True Residual	Computational Time
Matlab GMRES	38	1.5592×10^{-10}	4.5350×10^{-13}	$0.121s$
My GMRES	39	2.3797×10^{-13}	7.1893×10^{-14}	$4.943s$

Table 7: Results for preconditioned each GMRES implementation

There are clear differences in the residuals and computation time values. My implementation is 40 times slower but produces a true residual 5 times smaller.

With L

Method	Iterations	Final Residual	Computational Time
GMRES	1	3.8481×10^{-16}	$0.027s$
My PCG	1	1.7554×10^{-17}	$0.003s$

Table 8: Iterations for each value of nx

GMRES without preconditioning, My PCG with L as identity matrix

Method	Iterations	Final Residual	Computational Time
GMRES	6	3.0413×10^{-14}	$0.102s$
My PCG	5	1.0468×10^{-13}	$0.012s$

Table 9: Iterations for each value of nx, no preconditioning

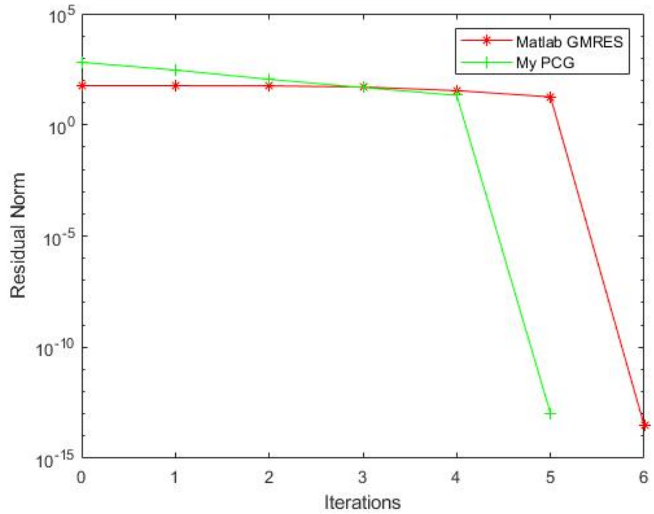

Figure 8: semilogy plot ex6, GMRES without preconditioning, My PCG with L as identity matrix

Question 7

Restart	Iterations	Final Residual	Computational Time
10	1149	9.6915×10^{-13}	$2.235s$
20	739	9.6140×10^{-13}	$1.443s$
30	88	6.7203×10^{-13}	$0.242s$
50	41	4.8414×10^{-13}	$0.135s$

Table 10: Iterations for each value of nx, no preconditioning

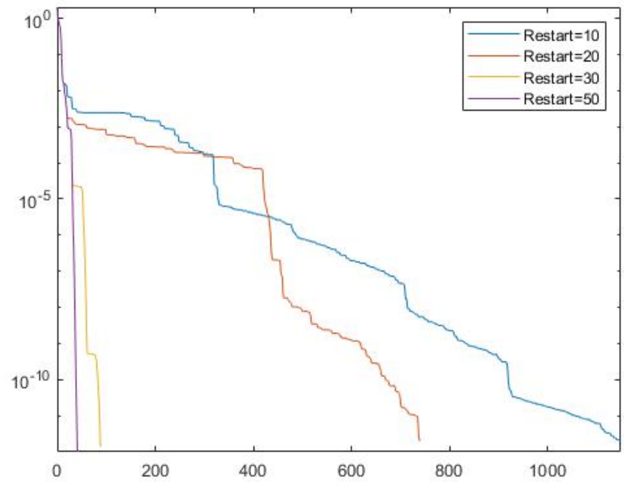

Figure 9: semilogy plot ex7

Question 8

Tolerance	Iterations	Tprec	Tsol	Ttotal	Final Residual	rho
2×10^{-2}	1983	$39.59s$	$77.65s$	$117.24s$	9.9053×10^{-13}	0.4537
1×10^{-2}	691	$36.46s$	$26.63s$	$63.09s$	9.7254×10^{-13}	0.5807
3×10^{-3}	247	$40.67s$	$11.04s$	$51.71s$	9.1709×10^{-13}	0.9401
1×10^{-3}	102	$37.61s$	$5.63s$	$43.24s$	8.7501×10^{-13}	1.4544
1×10^{-4}	34	$42.44s$	$2.93s$	$45.37s$	4.5169×10^{-13}	3.5140
1×10^{-5}	16	$76.50s$	$2.49s$	$78.99s$	4.9947×10^{-13}	9.0720

Table 11: Iterations for each value of nx, no preconditioning

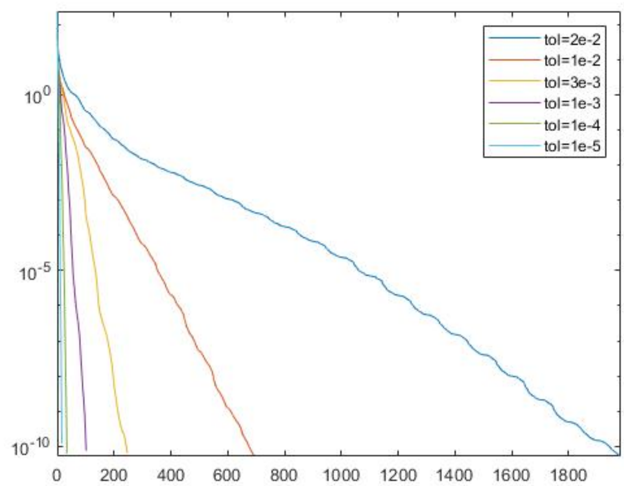

Figure 10: semilogy plot ex8