数值代数第2章上机实验报告

211840196 张博阳

摘要

本实验报告基于第2章所学的各解线性方程组的迭代法对上机实验题给出了相应的程序实现与执行情况，其中阐述了所使用的各迭代法的理论背景和实现以及代码实现上为提升精度和速度所使用的技术，并基于实际程序执行情况反馈对算法的实现结果及其性能进行了评价。

1 前言

解线性方程组的迭代法是在对于解线性方程组问题中在时间和精度上的折中解决方案，其一定程度上牺牲了解的精度，换来比直接法更快的计算速度和具有实际意义的中间结果。第2章上机实验题基于迭代法的这一特征给出了可利用迭代法求解的多个问题的解决方案。

2 数学原理和程序设计流程

考虑用迭代法解线性方程组$Ax=b$，其中$A=(a_{ij})_n, b=[b_1\ b_2\ \dots b_n]^{\mathrm{T}}$。

2.1 Jacobi迭代法

基于矩阵分裂$A=D-(D-A)$，其中$D=\mathrm{diag}\{a_{11},a_{22},\dots,a_{nn}\}$是$A$的主对角元素组成的对角阵，引出一阶迭代方法

$$\begin{aligned}x^{(k)}&=Bx^{(k-1)}+g\\&=(I-D^{-1}A)x^{(k-1)}+D^{-1}b\end{aligned}$$

其计算$x^{(k)}$各分量的计算式为

$$x_i^{(k)}=\frac{1}{a_{ii}}\left(b_i-\sum_{j=1,j\neq i}^{n}a_{ij}x_j^{(k-1)}\right),i=1,2,\dots,n,k\in\mathbb{N}^*$$

2.2 Gauss-Seidel迭代法

基于矩阵分裂$A=D-DL-DU$，其中$D=\mathrm{diag}\{a_{11},a_{22},\dots,a_{nn}\}$是$A$的主对角元素组成的对角阵，$-DL$是$A$的严格下三角部分，$-DU$是$A$的严格上三角部分，引出一阶迭代方法

$$\begin{aligned}x^{(k)}&=T_1x^{(k-1)}+g\\&=(I-L)^{-1}Ux^{(k-1)}+(I-L)^{-1}D^{-1}b\end{aligned}$$

亦即

$$x^{(k)} = Lx^{(k)} + Ux^{(k-1)} + D^{-1}b$$

其计算$x^{(k)}$各分量的计算式为

$$x_i^{(k)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij}x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij}x_j^{(k-1)}\right), i = 1, 2, \ldots, n, k \in \mathbb{N}^*$$

2.3 SOR方法

基于矩阵分裂$A = \frac{1}{\omega}(D - \omega DL) - \frac{1}{\omega}((1-\omega)D + \omega DU)$ $(\omega \neq 0)$，引出一阶迭代方法

$$\begin{aligned} x^{(k)} &= T_\omega x^{(k-1)} + g \\ &= (I - \omega L)^{-1}((1-\omega)I + \omega U)x^{(k-1)} + \omega(I - \omega L)^{-1}D^{-1}b \end{aligned}$$

可以证明ω存在最优参数选取

$$\omega = \frac{2}{1 + \sqrt{1 - \mu^2}}$$

其中μ是对应Jacobi方法的迭代矩阵矩阵B的特征值。

2.4 迭代加速方法（Chebyshev半迭代法）

定义修正序列

$$y^{(m)} = \sum_{k=0}^{m} a_{m,k}x^{(k)}$$

对$\{x^{(k)}\}$的迭代转变为对$\{y^{(k)}\}$的迭代，并调整方法使得序列$\{y^{(k)}\}$更快地收敛于真解x^*。对于一阶线性非定常迭代法

$$x^{(m)} = x^{(m-1)} + \tau_m(b - Ax^{(m-1)})$$

其被称为Richardson迭代法，适当选取参数组$\{\tau_m\}$，其可视为对应的一阶线性定常方法的半迭代加速。可以证明，一阶定常方法的最优参数设置为

$$\tau = \frac{2}{\lambda_{\max} + \lambda_{\min}}$$

其中$\lambda_{\max}, \lambda_{\min}$分别为$A$的最大和最小特征值，而给定总体迭代步数$m$，基于Chebyshev多项式给出的最优参数组$\{\tau_k\}_{k=1}^m$为

$$\tau_k = \frac{1}{\frac{\lambda_{\max} - \lambda_{\min}}{2}\cos\left(\frac{2k-1}{2m}\pi\right) + \frac{\lambda_{\max} + \lambda_{\min}}{2}}$$

当定常迭代法的迭代矩阵G为实对称正定矩阵时，可以给出对应的使用Chebyshev半迭代加速技术得到的非定常二阶迭代法

$$\begin{aligned} y^{(k+1)} &= \frac{2T_k(\xi)l(G)y^{(k)}}{T_{k+1}(\xi)} - \frac{T_{k-1}(\xi)y^{(k-1)}}{T_{k+1}(\xi)} + \frac{4}{\lambda_{\max} - \lambda_{\min}}\frac{T_k(\xi)g}{T_{k+1}(\xi)} \\ &= \rho_{k+1}\left[\nu\left(Gy^{(k)} + g\right) + (1-\nu)y^{(k)}\right] + (1 - \rho_{k+1})\, y^{(k-1)} \end{aligned}$$

其中$\lambda_{\max}, \lambda_{\min}$分别为迭代矩阵$G$的最大和最小特征值，$\nu = \frac{2}{2 - \lambda_{\max} - \lambda_{\min}}, \xi = l(1) = \frac{2 - \lambda_{\max} - \lambda_{\min}}{\lambda_{\max} - \lambda_{\min}}, \rho_{k+1} = \frac{2\xi T_k(\xi)}{T_{k+1}(\xi)} = \frac{1}{1 - \frac{\rho_k}{4\xi^2}}, \rho_1 = 2$。

2.5 共轭斜量法

解线性方程组$Ax=b$等价于求解优化问题

$$x=\arg\min_{x\in\mathbb{R}^n}\frac{1}{2}x^{\mathrm{T}}Ax-b^{\mathrm{T}}x$$

故考虑构造求解优化问题的迭代序列$\{x^{(k)}\}$，其由如下从$x^{(k)}$出发，以p_k为搜索方向，α_k为步长的迭代式生成

$$x^{(k+1)}=x^{(k)}+\alpha_k p_k$$

以此为出发点构造的共轭斜量法为

$$p_0=-r_0=b-Ax^{(0)}$$

$$k=0,1,\ldots\begin{cases}\alpha_k=-\dfrac{r_k^{\mathrm{T}}p_k}{p_k^{\mathrm{T}}Ap_k}\\ x^{(k+1)}=x^{(k)}+\alpha_k p_k\\ r_{k+1}=Ax^{(k+1)}-b=r_k-\alpha_k Ap_k\\ \beta_k=\dfrac{r_{k+1}^{\mathrm{T}}Ap_k}{p_k^{\mathrm{T}}Ap_k}\\ p_{k+1}=-r_{k+1}+\beta_k p_k\end{cases}$$

可以证明

$$r_i^{\mathrm{T}}r_j=0,i\neq j$$

$$r_k^{\mathrm{T}}p_i=0,i=0,1,\ldots,k-1$$

$$p_k^{\mathrm{T}}r_i=-r_k^{\mathrm{T}}r_k,i=0,1,\ldots,k$$

出于减少计算步数的考虑，共轭斜量法可以改写为

$$p_0=-r_0=b-Ax_0$$

$$k=0,1,\ldots\begin{cases}\alpha_k=\dfrac{r_k^{\mathrm{T}}r_k}{p_k^{\mathrm{T}}Ap_k}\\ x^{(k+1)}=x^{(k)}+\alpha_k p_k\\ r_{k+1}=Ax_{k+1}-b=r_k-\alpha_k Ap_k\\ \beta_k=\dfrac{r_{k+1}^{\mathrm{T}}r_{k+1}}{r_k^{\mathrm{T}}r_k}\\ p_{k+1}=-r_{k+1}+\beta_k p_k\end{cases}$$

$r_k^{\mathrm{T}}r_k$在每次循环时可以重复使用，只需算一个$r_k^{\mathrm{T}}r_k$和$p_k^{\mathrm{T}}Ap_k$即可。

2.6 针对问题的程序设计流程

2.6.1 练习6.2.1

程序分别采用残量准则、相邻误差准则、后验误差准则和1-范数、无穷范数计算n取不同的值时Jacobi方法和Gauss-Seidel方法解线性方程组达到用户指标$\epsilon=10^{-6}$所需迭代步数以及对应停机时的真实误差$\|x-x^*\|$。主函数`work_621.m`调用封装了用Jacobi方法解线性方程组的函数`Jacobi.m`和Gauss-Seidel方法的函数`Gauss_Seidel.m`，通过改变传入参数调整方法中所使用的停机准则和范数。

2.6.2 练习6.2.2

程序封装SOR方法在函数`SOR.m`中，采用真实误差的Euclid范数作为停机标准，以0.05为步长在区间(0,2)中搜索最佳松弛因子。

2.6.3 练习6.2.3

程序基于练习6.2.1中已封装的函数`Jacobi.m`、`Gauss_Seidel.m`和练习6.2.2中已封装的`SOR.m`传入最佳松弛因子作为`omega`参数封装函数`best_parameter_SOR.m`设计了整体程序框架。

2.6.4 练习6.2.4

程序采用Euclid范数作为所用范数，将变系数R方法封装在函数`best_parameter_Richardson.m`中，根据理论分析，m过大时会出现”反弹“现象，因此应选取较大的m取值范围以观察m超过最佳取值后算法的数值表现。

2.6.5 练习6.2.5

程序将半迭代加速的Jacobi方法封装在函数`iterative_Jacobi.m`中。对于$m = +\infty$的情形，设置了用户指标$\epsilon = 10^{-6}$并采用了残量准则作为停机准则。

2.6.6 练习6.2.6

采用残量准则和Euclid范数的CG方法对应伪代码如下。

Algorithm 1 CG算法

Input: 系数矩阵A，右端向量b，最大迭代步数n_0，用户指标ϵ

Initialize: 迭代步数记录n，初始向量x_0

$r \leftarrow Ax_0 - b$
$p \leftarrow -r$
$k_1 \leftarrow r^{\mathrm{T}} r$
for $n = 1 : n_0$ **do**
  $k_2 \leftarrow p^{\mathrm{T}} Ap$
  $\alpha \leftarrow \dfrac{k_1}{k_2}$
  $x \leftarrow x_0 + \alpha p$
  if $\|Ax - b\|_2 < \epsilon$ **then**
    退出循环
  else
    $x_0 \leftarrow x$
    $r \leftarrow r + \alpha Ap$
    $k_3 \leftarrow r^{\mathrm{T}} r$
    $\beta \leftarrow \dfrac{k_3}{k_1}$
    $p \leftarrow -r + \beta p$
    $k_1 \leftarrow k_3$
  end if

18: **end for**
19: **Output:** 达到用户指标的线性方程组$Ax = b$的解x和此时迭代步数n

程序采用k_1, k_2, k_3三个中间变量记录中间值，减少了重复的内积计算。

3 实验结果和数据讨论

3.1 练习6.2.1

主程序`work_621.m`的运行结果如Figure 1-12所示。

就每个算法而言，停机算法和真实误差在$n > 35$后均快速上升，并且换用不同的停机准则和向量范数在数值表现上有较大的差异。Jacobi方法在使用后验误差准则和无穷范数时出现了较大的波动，方法不再收敛于线性方程组的真解。

算法间对比上，Gauss-Seidel方法的迭代次数大约是Jacobi方法的一半，并且真实误差在大多数情况下表现优于Jacobi方法。

3.2 练习6.2.2

取$n = 22, 20, 15, 24, 11$，得到的最佳松弛因子ω扫描值分别为$1.7500, 1.7500, 1.6500, 1.7500, 1.6000$，理论计算结果为$1.7603, 1.7406, 1.6735, 1.7773, 1.5888$，符合理论估计和误差理论分析。

3.3 练习6.2.3

主程序`work_623.m`的运行结果如Figure 13-15所示。

从图中可以看出，具有最优参数的SOR方法表现远好于Jacobi方法和Gauss-Seidel算法，其在$6 \leq n \leq 40$范围内数值误差保持相对稳定，而残量随n的增大整体呈现越来越小的趋势。迭代步数在$n = 40$时为169，远小于Jacobi方法的4381和Gauss-Seidel方法的2192，其与n大致呈三次关系（R^2值分别为1、1、0.9999）。

3.4 练习6.2.4

主程序`work_624.m`的运行结果如Figure 16-17所示，程序选取$n = 50$，考察了$1 \leq m \leq 100$范围内变系数R方法的数值表现。

在误差和残量两个图线的变化来看，最优循环指标m的选取在30−35之间，在到达最优循环指标之前，误差和残量稳定下降，一旦超过最优循环指标即出现“反弹”现象，误差和残量大幅上升（$m = 100$时二者Euclid范数均超过10^{30}），这表明在选择循环指标m时要事先作估计，否则变系数R方法不能迭代出线性方程组的解。

3.5 练习6.2.5

主程序`work_625.m`的运行结果如Figure 18-19所示。

由于Jacobi方法的半迭代加速的最优参数与循环指标m无关，从而若半迭代加速方法是相容的，那么循环指标m越大，迭代次数越多，其在面对n较大的情形表现自然更好。从实际数值表现来看，$m = 25, 50, 100$随n的增加先后崩溃，而$m = +\infty$在实验范围内$(6 \leq n \leq 50)$的范围内依然能够给出误差的Euclid范数小于10^{-4}的解。

3.6 练习6.2.6

主程序`work_626.m`的运行结果如Figure 20-22所示。

在Figure 22中可以看到，具有最优参数的SOR方法和CG算法迭代次数和n近似呈线性关系（线性相关系数分别为0.9981和0.9996），而CG算法所需迭代次数大约为最优参数的SOR方法的40%。

4 小结

通过上述实验题，不同解线性方程组的迭代法在同一问题下的数值表现得到了比较。诸如Jacobi方法和Gauss-Seidel方法等古典算法在面对$n = 50$时计算速度依然不理想，但如带有最佳松弛因子的SOR算法和CG算法的速度已经远远超出使用直接法解相同阶数的线性方程组，并且中间过程具有意义，可以暂时停机并将迭代结果作为下次执行程序的初始向量。

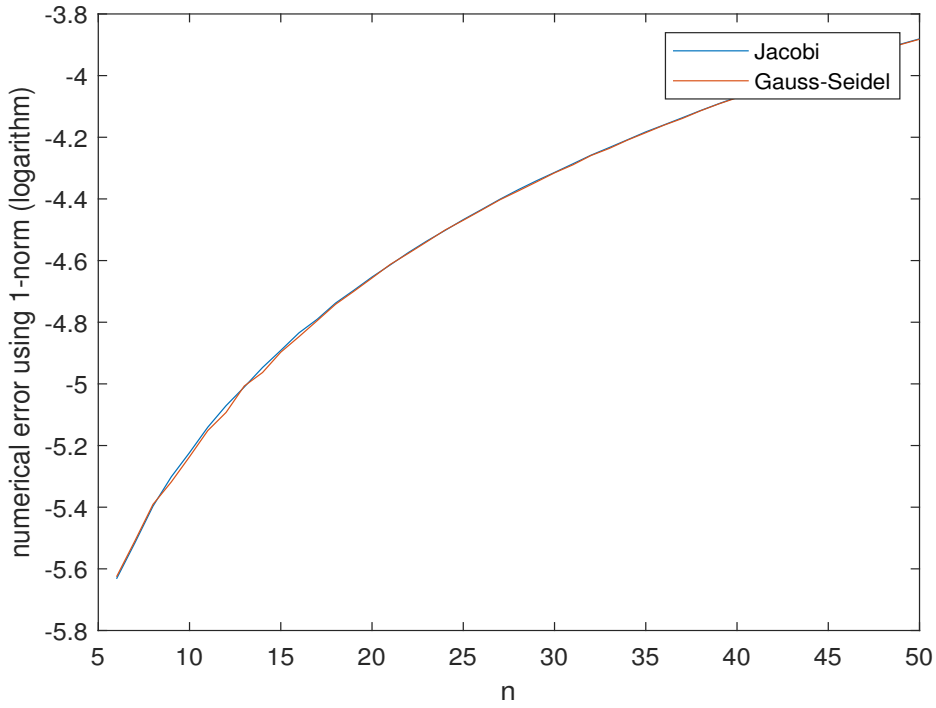

Figure 1: The numerical error using the remaining rule and 1-norm

参考文献

[1]林成森. 数值计算方法(下册)[M]. 北京: 科学出版社, 2005.

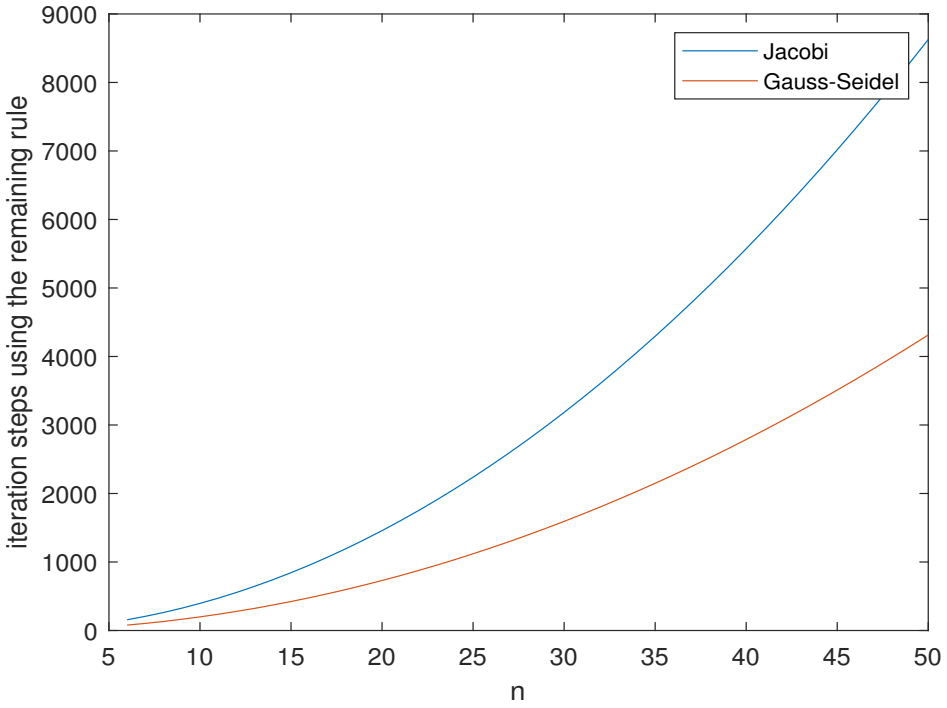

Figure 2: The iteration steps using the remaining rule and 1-norm

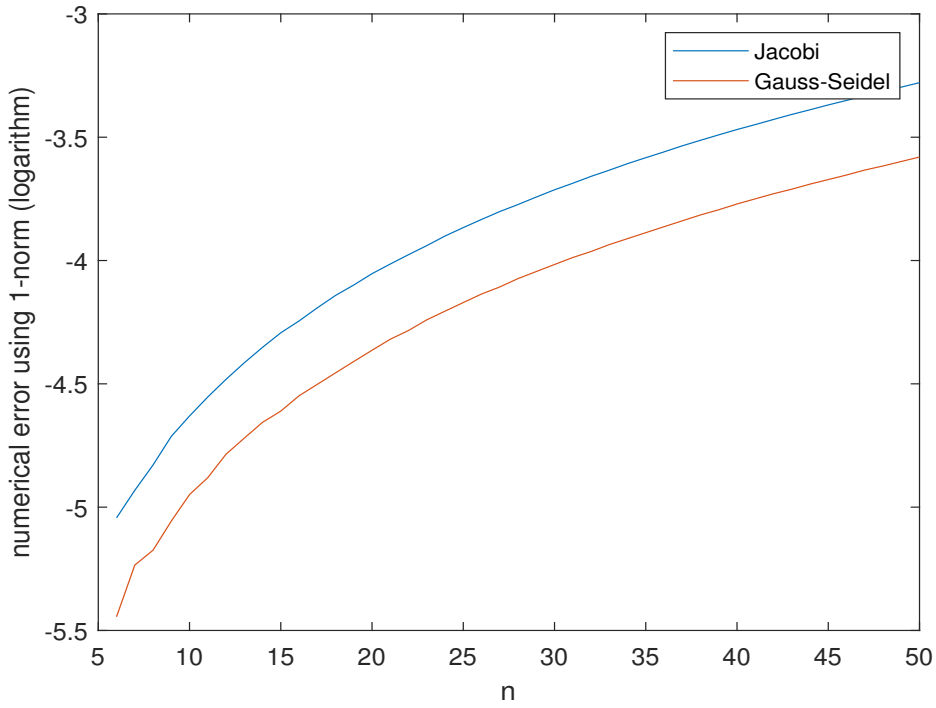

Figure 3: The numerical error using the adjacent rule and 1-norm

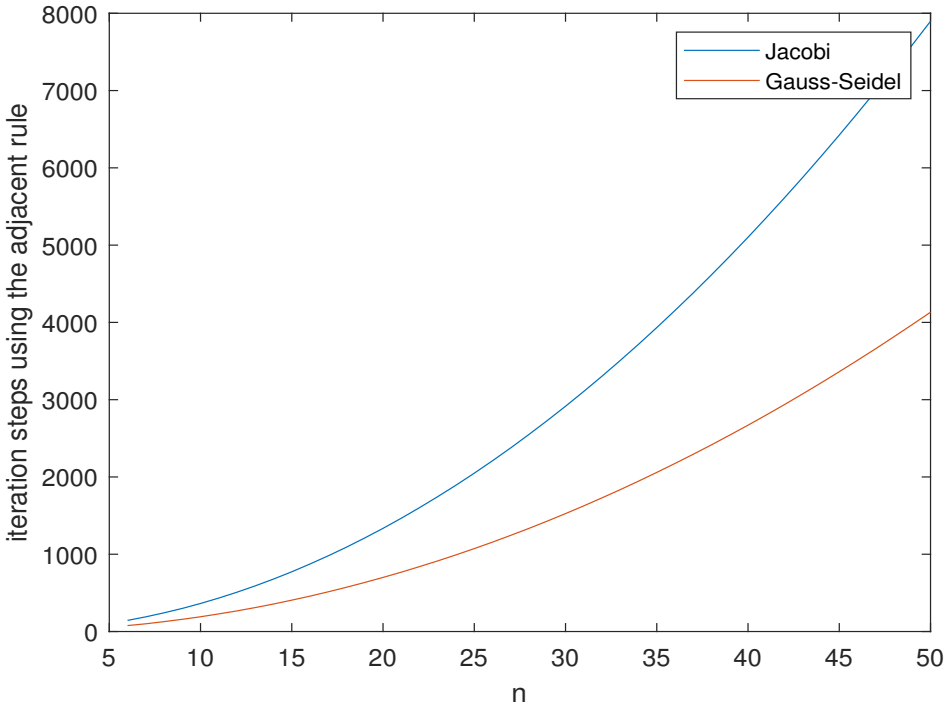

Figure 4: The iteration steps using the adjacent rule and 1-norm

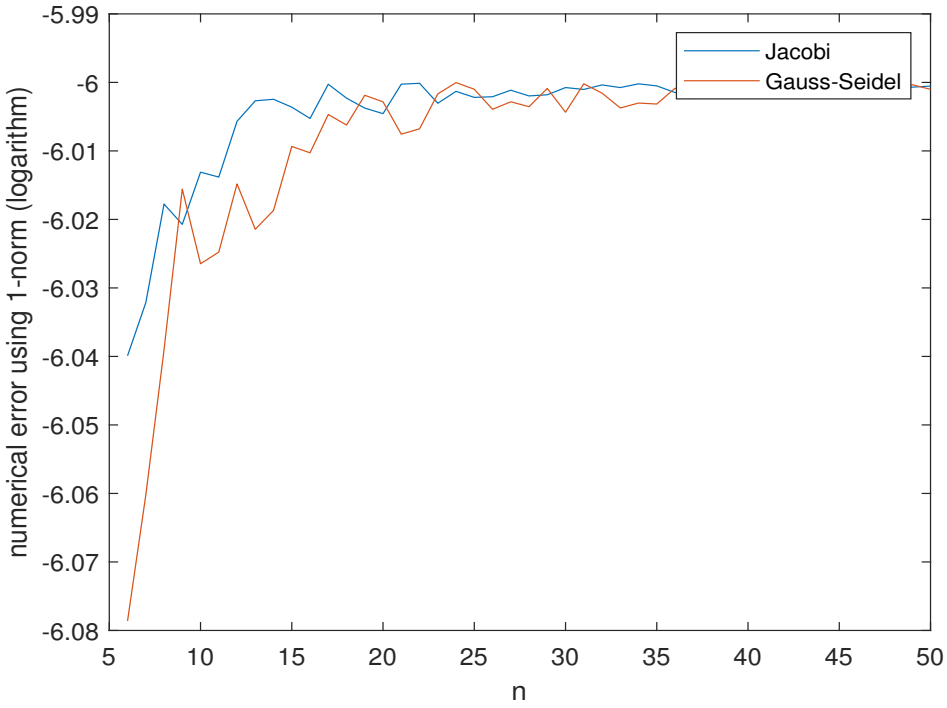

Figure 5: The numerical error using the backward rule and 1-norm

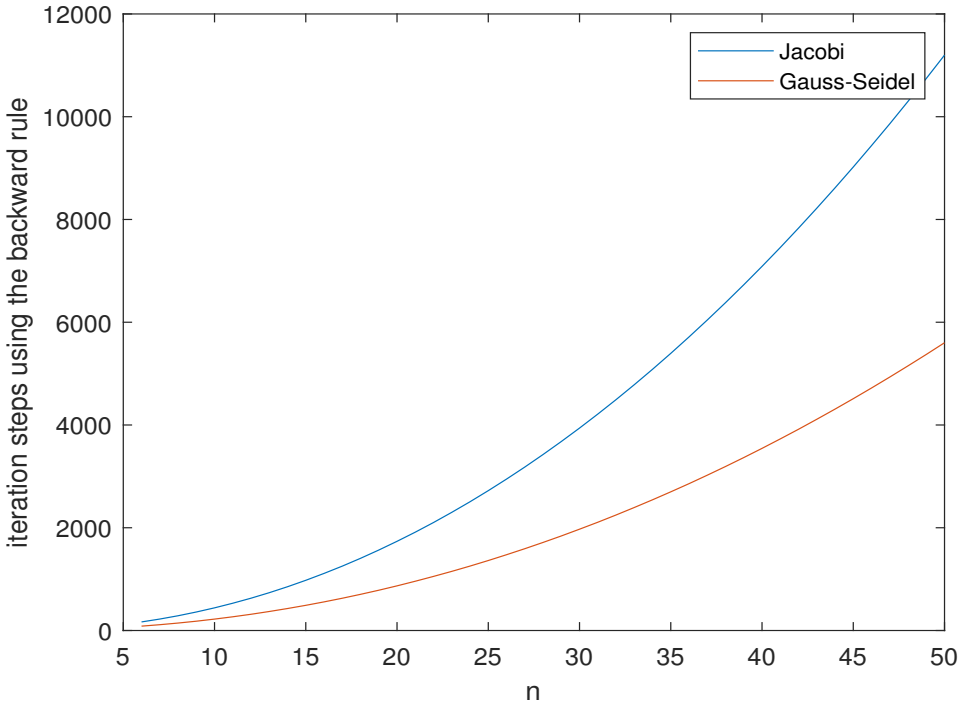

Figure 6: The iteration steps using the backward rule and 1-norm

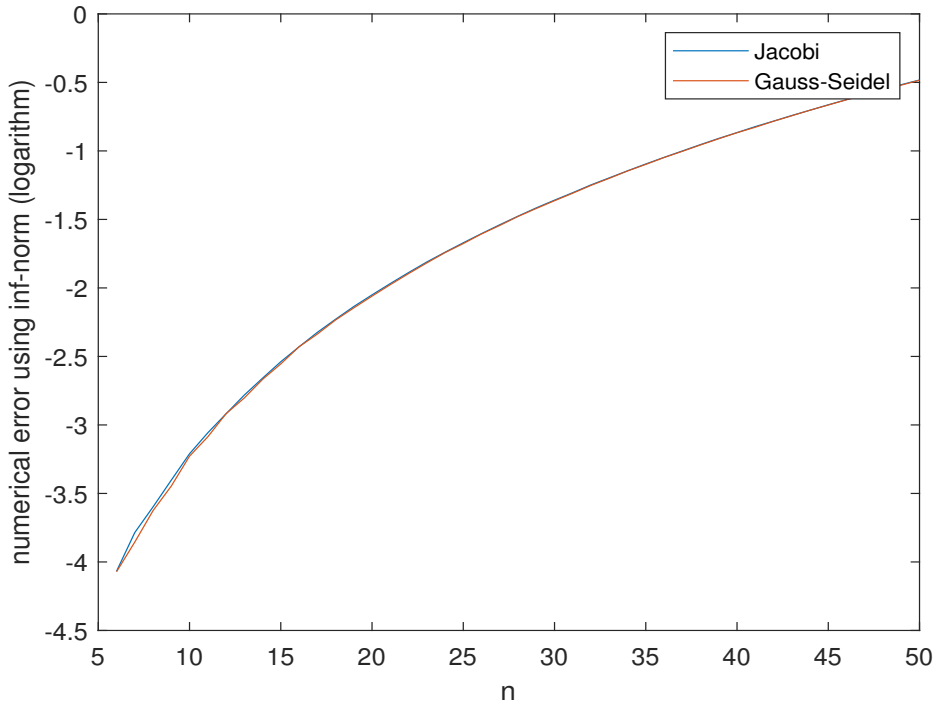

Figure 7: The numerical error using the remaining rule and inf-norm

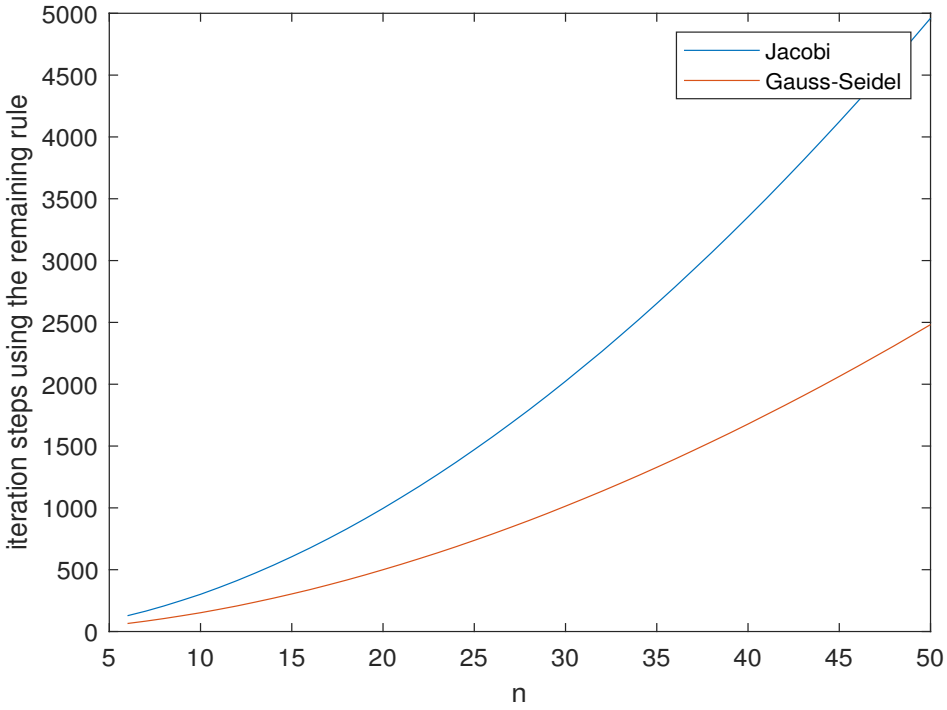

Figure 8: The iteration steps using the remaining rule and inf-norm

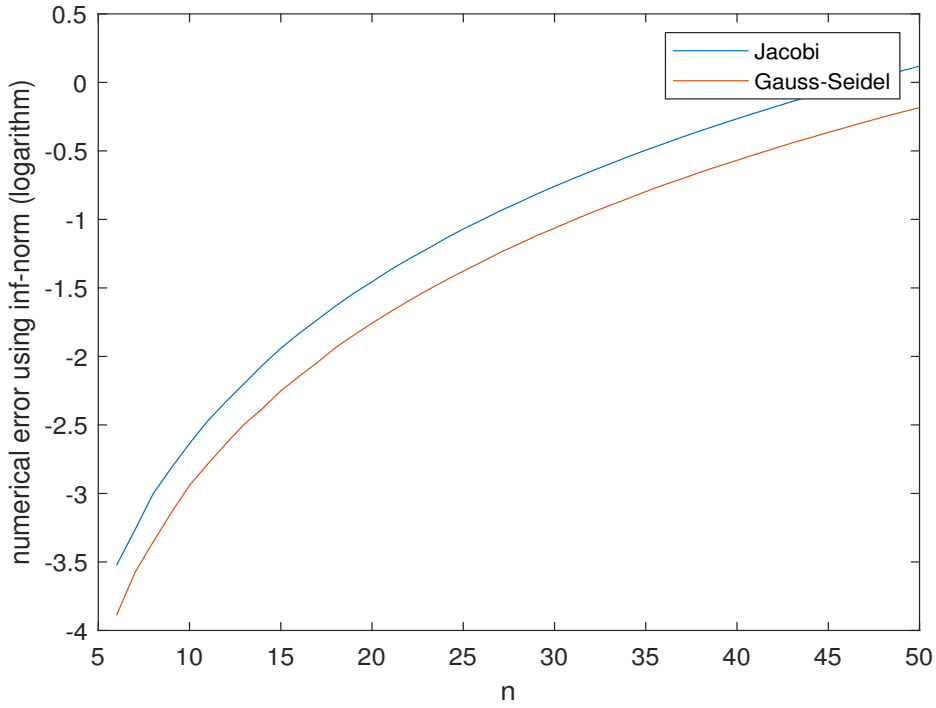

Figure 9: The numerical error using the adjacent rule and inf-norm

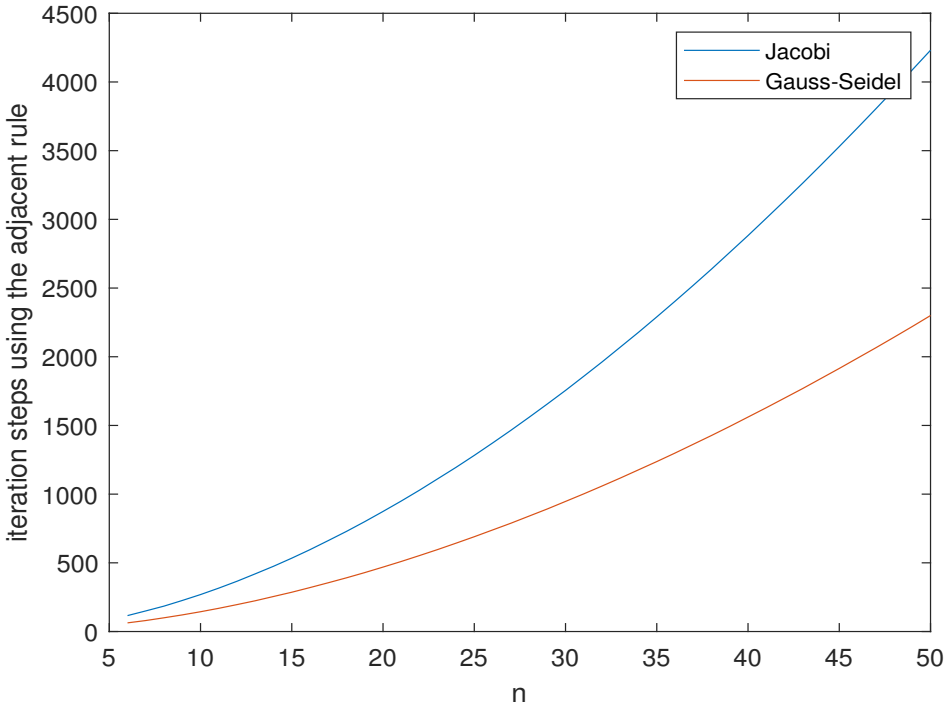

Figure 10: The iteration steps using the adjacent rule and inf-norm

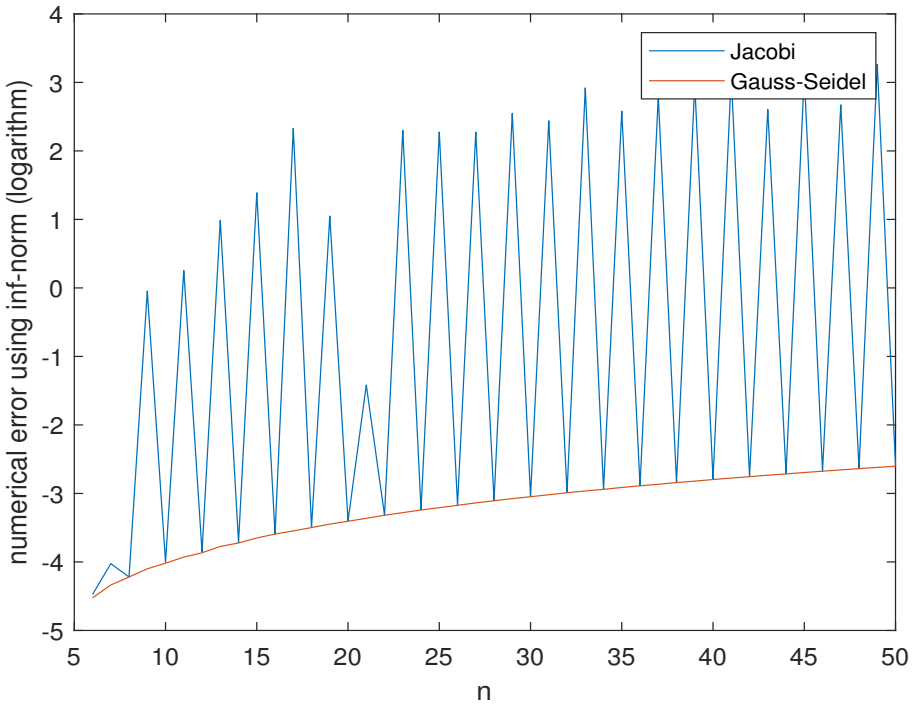

Figure 11: The numerical error using the backward rule and inf-norm

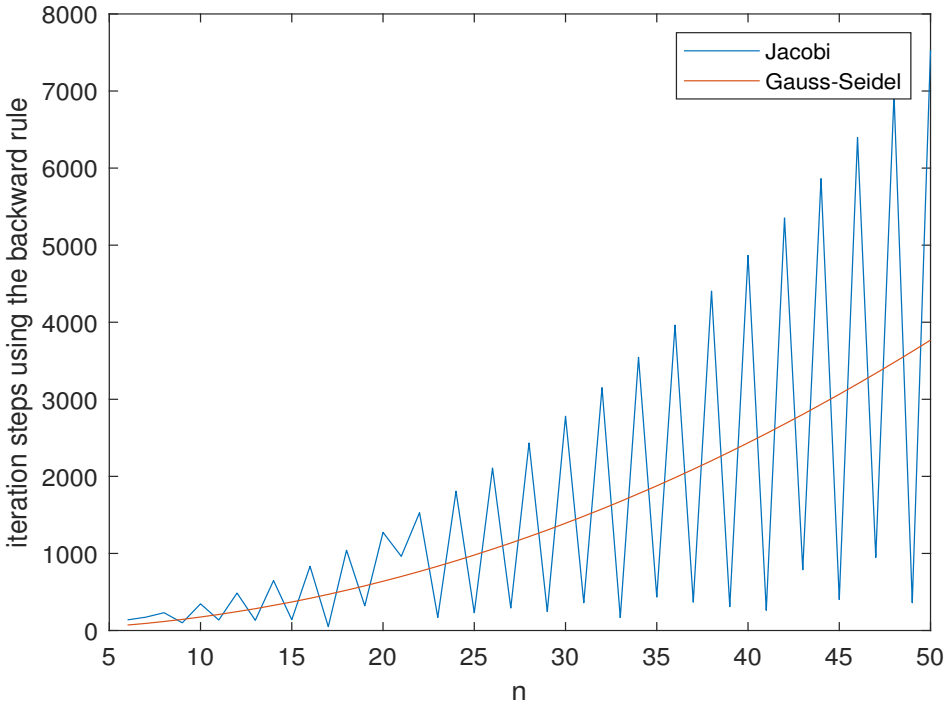

Figure 12: The iteration steps using the backward rule and inf-norm

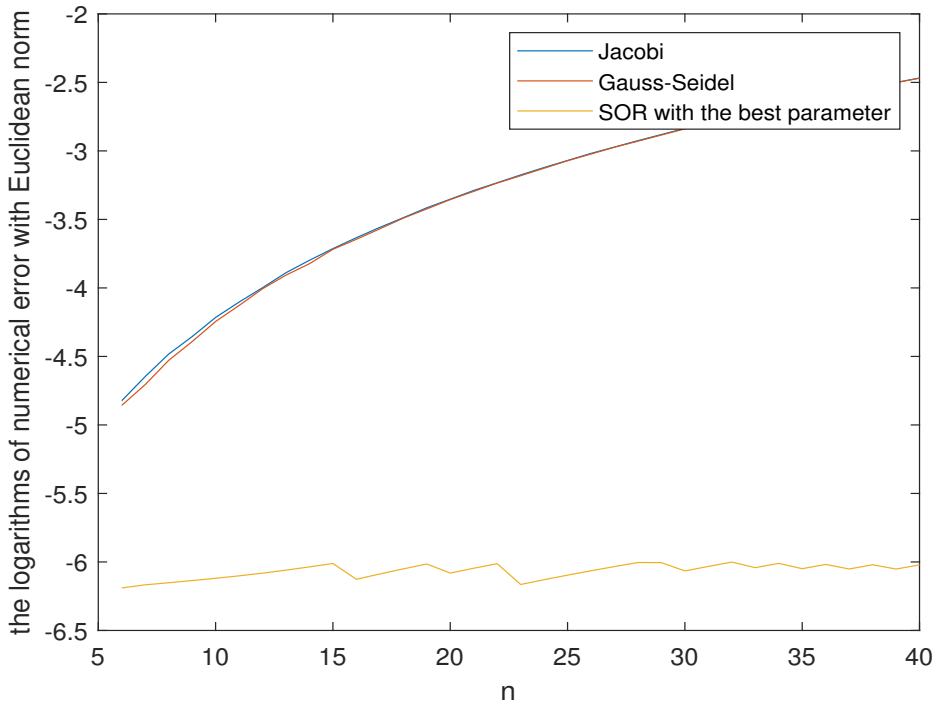

Figure 13: The numerical error of the 3 algorithms

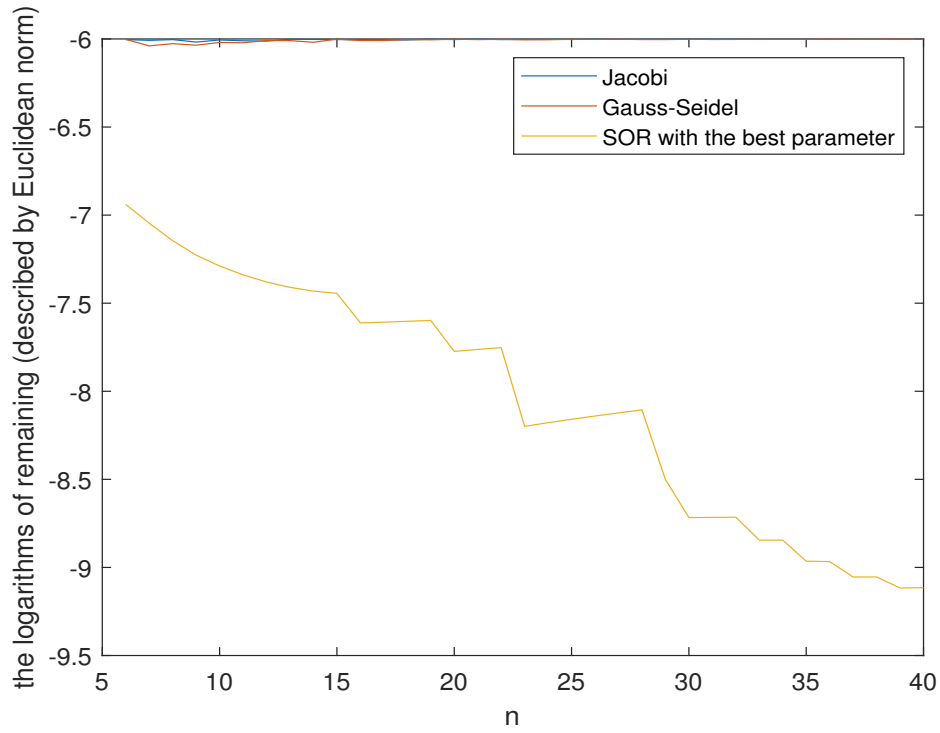

Figure 14: The remaining of the 3 algorithms

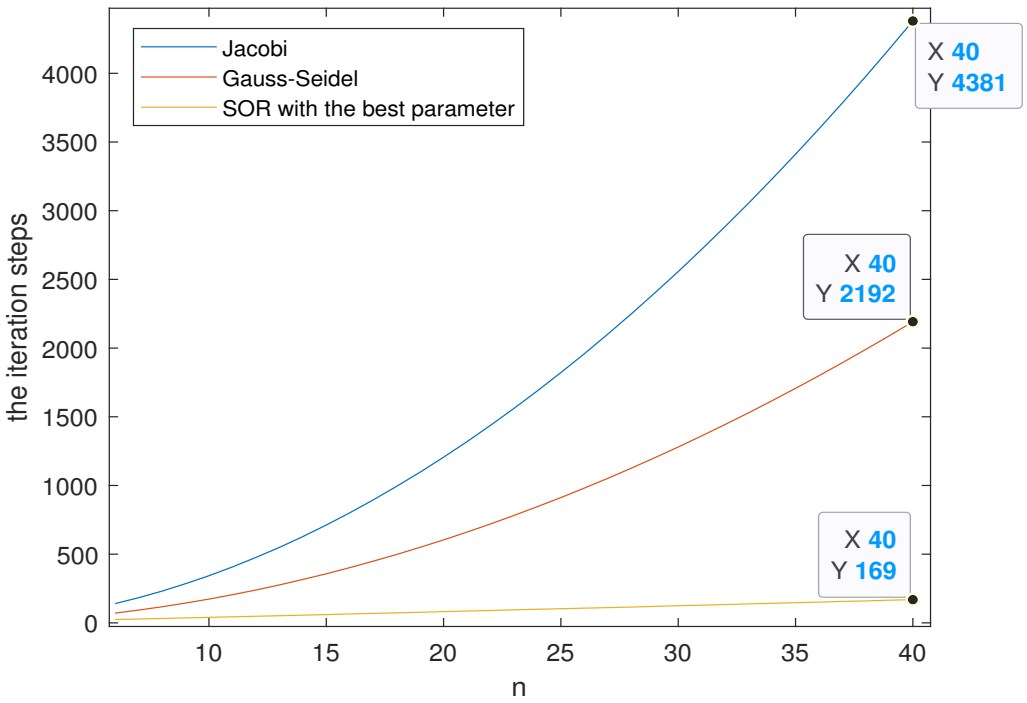

Figure 15: The iteration steps using the 3 algorithms

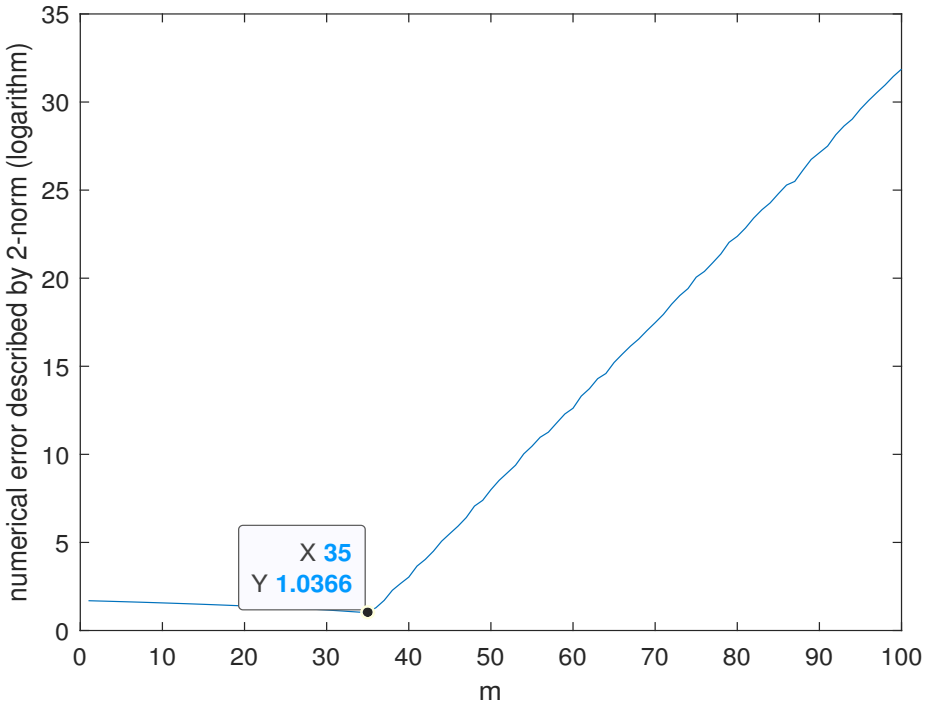

Figure 16: The numerical error trend by changing m

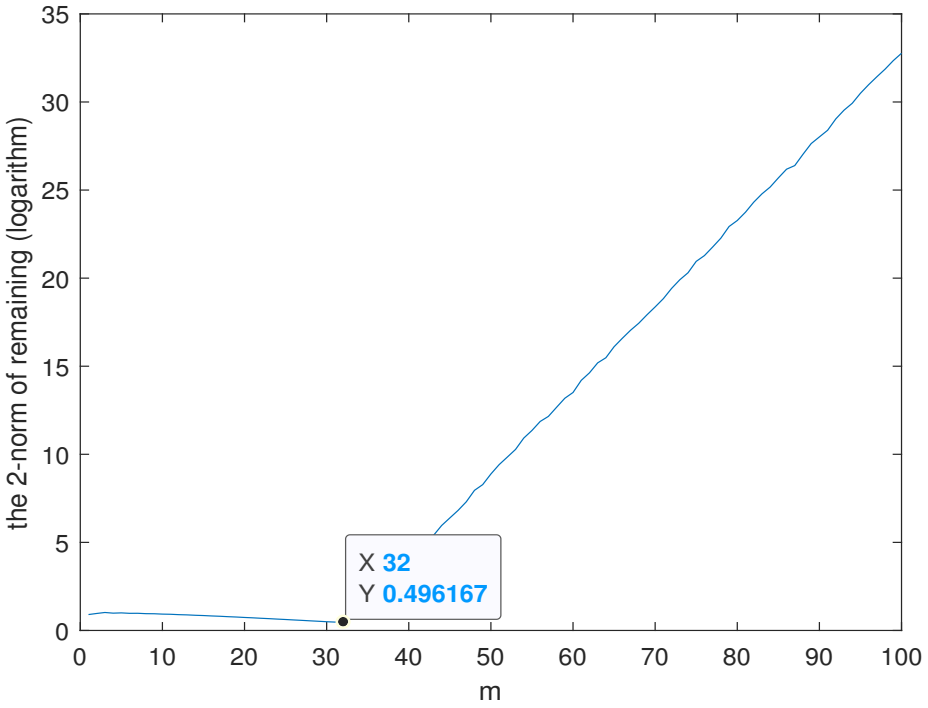

Figure 17: The remaining of different m

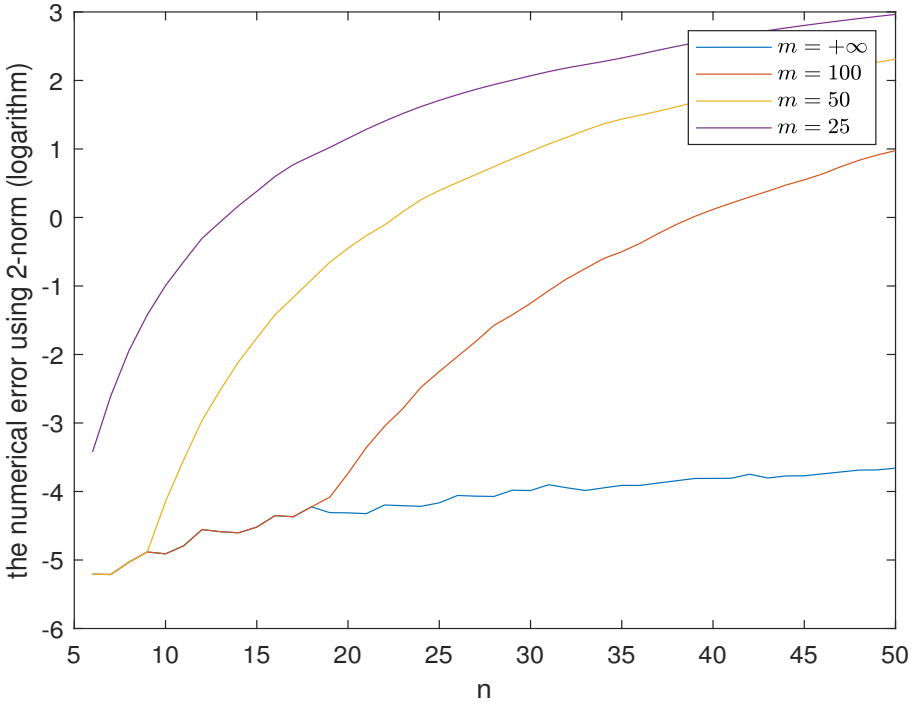

Figure 18: The numerical trend of different m

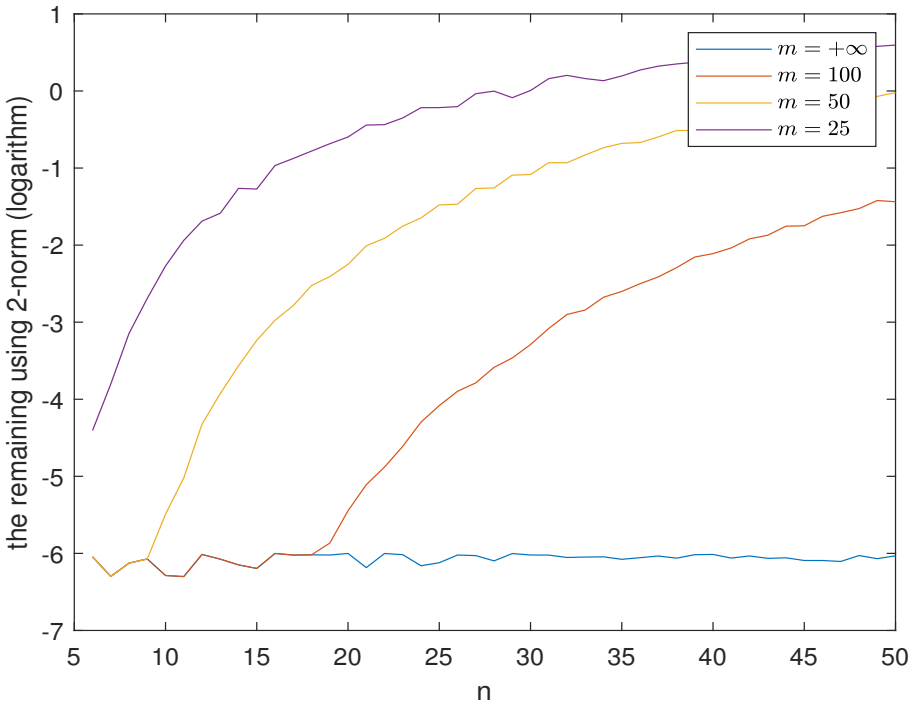

Figure 19: The remaining trend by changing m

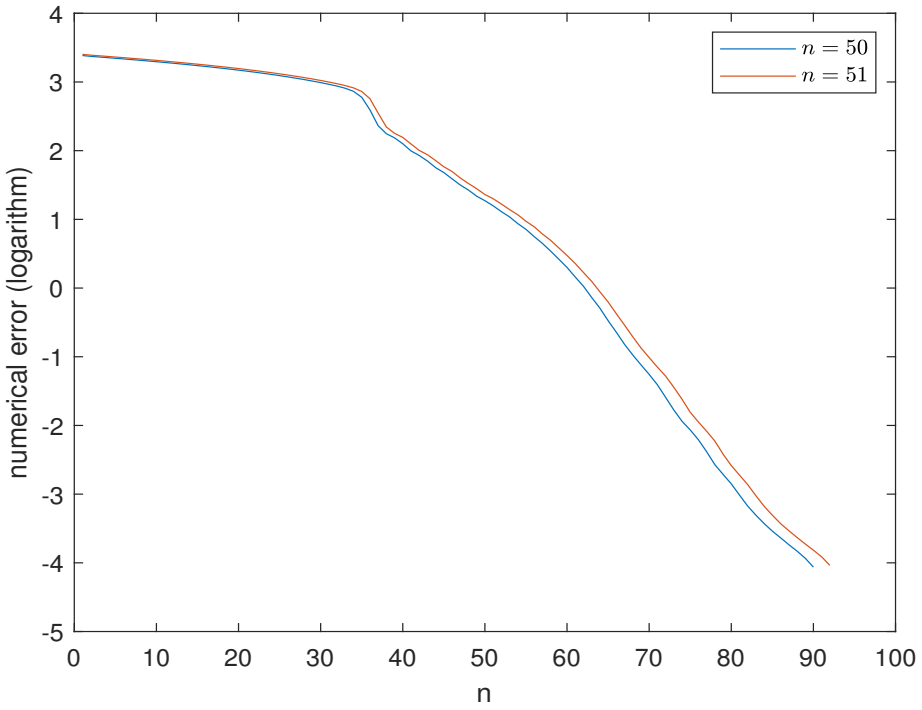

Figure 20: The numerical error in the case of n=50 and n=51 as iterating

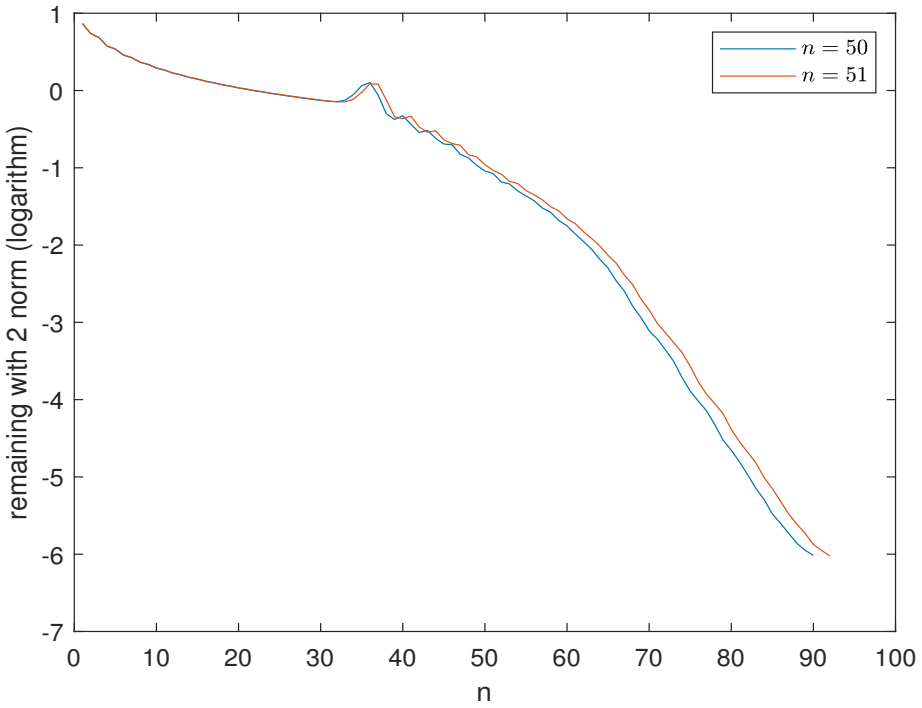

Figure 21: The remaining in the case of n=50 and n=51 as iterating

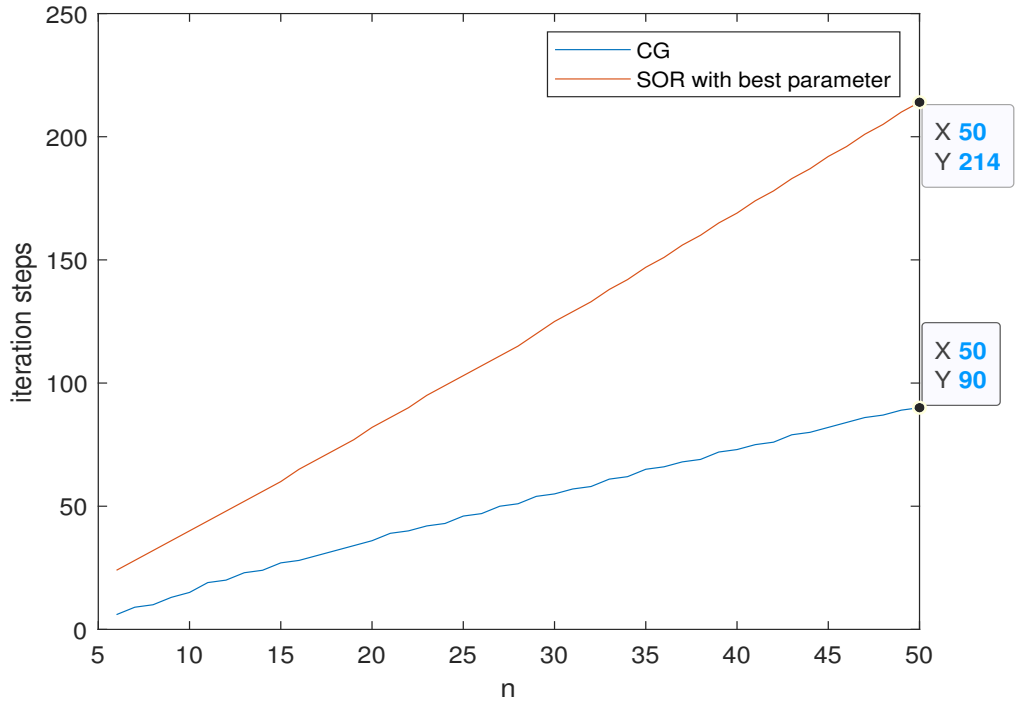

Figure 22: The iteration steps of CG method and SOR method with the best parameter